

autogluon_each_location

October 16, 2023

```
[1]: # config

label = 'y'
metric = 'mean_absolute_error'
time_limit = 60*5
presets = 'best_quality'

do_drop_ds = True
# hour, dayofweek, dayofmonth, month, year
use_dt_attrs = ["hour", "year"]
use_estimated_diff_attr = False
use_is_estimated_attr = True

use_groups = False
n_groups = 8

auto_stack = True
num_stack_levels = 1
num_bag_folds = 0
if auto_stack:
    num_stack_levels = None
    num_bag_folds = None

use_tune_data = False
use_test_data = True
tune_and_test_length = 24*30*3 # 3 months from end
holdout_frac = None
use_bag_holdout = False # Enable this if there is a large gap between score_val_
    ↪ and score_test in stack models.

sample_weight = None #'sample_weight' #None
weight_evaluation = False
sample_weight_estimated = 1

run_analysis = True
```

```

[2]: import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings("ignore")

def feature_engineering(X):
    # shift all columns with "1h" in them by 1 hour, so that for index 16:00,
    ↪ we have the values from 17:00
    # but only for the columns with "1h" in the name
    #X_shifted = X.filter(regex="\dh").shift(-1, axis=1)
    #print(f"Number of columns with 1h in name: {X_shifted.columns}")

    columns = ['clear_sky_energy_1h:J', 'diffuse_rad_1h:J', 'direct_rad_1h:J',
               'fresh_snow_12h:cm', 'fresh_snow_1h:cm', 'fresh_snow_24h:cm',
               'fresh_snow_3h:cm', 'fresh_snow_6h:cm']

    X_shifted = X[X.index.minute==0][columns].shift(-1, axis=0)
    X_old_unshifted = X[X.index.minute==0][columns]
    # rename X_old_unshifted columns to have _not_shifted at the end
    X_old_unshifted.columns = [f"{col}_not_shifted" for col in X_old_unshifted.
    ↪ columns]

    # put the shifted columns back into the original dataframe
    #X[columns] = X_shifted[columns]

    date_calc = None
    if "date_calc" in X.columns:
        date_calc = X[X.index.minute == 0]['date_calc']

    # resample to hourly
    X = X.resample('H').mean()

    X[columns] = X_shifted[columns]
    X[X_old_unshifted.columns] = X_old_unshifted

    if date_calc is not None:
        X['date_calc'] = date_calc

    return X

```

```

def fix_X(X, name):
    # Convert 'date_forecast' to datetime format and replace original column
    # with 'ds'
    X['ds'] = pd.to_datetime(X['date_forecast'])
    X.drop(columns=['date_forecast'], inplace=True, errors='ignore')
    X.sort_values(by='ds', inplace=True)
    X.set_index('ds', inplace=True)

    X = feature_engineering(X)

    return X

def handle_features(X_train_observed, X_train_estimated, X_test, y_train):
    X_train_observed = fix_X(X_train_observed, "X_train_observed")
    X_train_estimated = fix_X(X_train_estimated, "X_train_estimated")
    X_test = fix_X(X_test, "X_test")

    # add sample weights, which are 1 for observed and 3 for estimated
    X_train_observed["sample_weight"] = 1
    X_train_estimated["sample_weight"] = sample_weight_estimated
    X_test["sample_weight"] = sample_weight_estimated

    y_train['ds'] = pd.to_datetime(y_train['time'])
    y_train.drop(columns=['time'], inplace=True)
    y_train.sort_values(by='ds', inplace=True)
    y_train.set_index('ds', inplace=True)

    return X_train_observed, X_train_estimated, X_test, y_train

def preprocess_data(X_train_observed, X_train_estimated, X_test, y_train,
                    location):
    # convert to datetime
    X_train_observed, X_train_estimated, X_test, y_train =
    handle_features(X_train_observed, X_train_estimated, X_test, y_train)

    if use_estimated_diff_attr:

```

```

X_train_observed["estimated_diff_hours"] = 0
X_train_estimated["estimated_diff_hours"] = (X_train_estimated.index -
↳pd.to_datetime(X_train_estimated["date_calc"])).dt.total_seconds() / 3600
X_test["estimated_diff_hours"] = (X_test.index - pd.
↳to_datetime(X_test["date_calc"])).dt.total_seconds() / 3600

X_train_estimated["estimated_diff_hours"] =
↳X_train_estimated["estimated_diff_hours"].astype('int64')
    # the filled once will get dropped later anyways, when we drop y nans
X_test["estimated_diff_hours"] = X_test["estimated_diff_hours"].
↳fillna(-50).astype('int64')

if use_is_estimated_attr:
    X_train_observed["is_estimated"] = 0
    X_train_estimated["is_estimated"] = 1
    X_test["is_estimated"] = 1

# drop date_calc
X_train_estimated.drop(columns=['date_calc'], inplace=True)
X_test.drop(columns=['date_calc'], inplace=True)

y_train["y"] = y_train["pv_measurement"].astype('float64')
y_train.drop(columns=['pv_measurement'], inplace=True)
X_train = pd.concat([X_train_observed, X_train_estimated])

# clip all y values to 0 if negative
y_train["y"] = y_train["y"].clip(lower=0)

X_train = pd.merge(X_train, y_train, how="inner", left_index=True,
↳right_index=True)

# print number of nans in sample_weight
print(f"Number of nans in sample_weight: {X_train['sample_weight'].isna().
↳sum()}")
# print number of nans in y
print(f"Number of nans in y: {X_train['y'].isna().sum()}")

X_train["location"] = location
X_test["location"] = location

return X_train, X_test
# Define locations
locations = ['A', 'B', 'C']

```

```

X_trains = []
X_tests = []
# Loop through locations
for loc in locations:
    print(f"Processing location {loc}...")
    # Read target training data
    y_train = pd.read_parquet(f'{loc}/train_targets.parquet')

    # Read estimated training data and add location feature
    X_train_estimated = pd.read_parquet(f'{loc}/X_train_estimated.parquet')

    # Read observed training data and add location feature
    X_train_observed = pd.read_parquet(f'{loc}/X_train_observed.parquet')

    # Read estimated test data and add location feature
    X_test_estimated = pd.read_parquet(f'{loc}/X_test_estimated.parquet')

    # Preprocess data
    X_train, X_test = preprocess_data(X_train_observed, X_train_estimated,
    ↪X_test_estimated, y_train, loc)

    X_trains.append(X_train)
    X_tests.append(X_test)

# Concatenate all data and save to csv
X_train = pd.concat(X_trains)
X_test = pd.concat(X_tests)

```

```

Processing location A...
Number of nans in sample_weight: 0
Number of nans in y: 0
Processing location B...
Number of nans in sample_weight: 0
Number of nans in y: 4
Processing location C...
Number of nans in sample_weight: 0
Number of nans in y: 6059

```

1 Feature engineering

```

[3]: import numpy as np
import pandas as pd

X_train.dropna(subset=['y'], inplace=True)

```

```

for attr in use_dt_attrs:
    X_train[attr] = getattr(X_train.index, attr)
    X_test[attr] = getattr(X_test.index, attr)

print(X_train.head())

if use_groups:
    # fix groups for cross validation
    locations = X_train['location'].unique() # Assuming 'location' is the name
    ↪ of the column representing locations

    grouped_dfs = [] # To store data frames split by location

    # Loop through each unique location
    for loc in locations:
        loc_df = X_train[X_train['location'] == loc]

        # Sort the DataFrame for this location by the time column
        loc_df = loc_df.sort_index()

        # Calculate the size of each group for this location
        group_size = len(loc_df) // n_groups

        # Create a new 'group' column for this location
        loc_df['group'] = np.repeat(range(n_groups),
    ↪ repeats=[group_size]*(n_groups-1) + [len(loc_df) - group_size*(n_groups-1)])

        # Append to list of grouped DataFrames
        grouped_dfs.append(loc_df)

    # Concatenate all the grouped DataFrames back together
    X_train = pd.concat(grouped_dfs)
    X_train.sort_index(inplace=True)
    print(X_train["group"].head())

to_drop = ["snow_drift:idx", "snow_density:kgm3"]

X_train.drop(columns=to_drop, inplace=True)
X_test.drop(columns=to_drop, inplace=True)

```

```
X_train.to_csv('X_train_raw.csv', index=True)
X_test.to_csv('X_test_raw.csv', index=True)
```

```

absolute_humidity_2m:gm3  air_density_2m:kgm3  \
ds
2019-06-02 22:00:00          7.700          1.22825
2019-06-02 23:00:00          7.700          1.22350
2019-06-03 00:00:00          7.875          1.21975
2019-06-03 01:00:00          8.425          1.21800
2019-06-03 02:00:00          8.950          1.21800

```

```

ceiling_height_agl:m  clear_sky_energy_1h:J  \
ds
2019-06-02 22:00:00      1728.949951          0.000000
2019-06-02 23:00:00      1689.824951          0.000000
2019-06-03 00:00:00      1563.224976          0.000000
2019-06-03 01:00:00      1283.425049      6546.899902
2019-06-03 02:00:00      1003.500000     102225.898438

```

```

clear_sky_rad:W  cloud_base_agl:m  dew_or_rime:idx  \
ds
2019-06-02 22:00:00          0.00      1728.949951          0.0
2019-06-02 23:00:00          0.00      1689.824951          0.0
2019-06-03 00:00:00          0.00      1563.224976          0.0
2019-06-03 01:00:00          0.75      1283.425049          0.0
2019-06-03 02:00:00         23.10      1003.500000          0.0

```

```

dew_point_2m:K  diffuse_rad:W  diffuse_rad_1h:J  ...  \
ds
2019-06-02 22:00:00      280.299988          0.000          0.000000  ...
2019-06-02 23:00:00      280.299988          0.000          0.000000  ...
2019-06-03 00:00:00      280.649994          0.000          0.000000  ...
2019-06-03 01:00:00      281.674988          0.300      7743.299805  ...
2019-06-03 02:00:00      282.500000         11.975     60137.601562  ...

```

```

direct_rad_1h:J_not_shifted  \
ds
2019-06-02 22:00:00          0.0
2019-06-02 23:00:00          0.0
2019-06-03 00:00:00          0.0
2019-06-03 01:00:00          0.0
2019-06-03 02:00:00          0.0

```

```

fresh_snow_12h:cm_not_shifted  \
ds
2019-06-02 22:00:00          0.0
2019-06-02 23:00:00          0.0
2019-06-03 00:00:00          0.0

```

2019-06-03 01:00:00	0.0
2019-06-03 02:00:00	0.0

fresh_snow_1h:cm_not_shifted \	
ds	
2019-06-02 22:00:00	0.0
2019-06-02 23:00:00	0.0
2019-06-03 00:00:00	0.0
2019-06-03 01:00:00	0.0
2019-06-03 02:00:00	0.0

fresh_snow_24h:cm_not_shifted \	
ds	
2019-06-02 22:00:00	0.0
2019-06-02 23:00:00	0.0
2019-06-03 00:00:00	0.0
2019-06-03 01:00:00	0.0
2019-06-03 02:00:00	0.0

fresh_snow_3h:cm_not_shifted \	
ds	
2019-06-02 22:00:00	0.0
2019-06-02 23:00:00	0.0
2019-06-03 00:00:00	0.0
2019-06-03 01:00:00	0.0
2019-06-03 02:00:00	0.0

fresh_snow_6h:cm_not_shifted sample_weight \		
ds		
2019-06-02 22:00:00	0.0	1
2019-06-02 23:00:00	0.0	1
2019-06-03 00:00:00	0.0	1
2019-06-03 01:00:00	0.0	1
2019-06-03 02:00:00	0.0	1

is_estimated y location			
ds			
2019-06-02 22:00:00	0	0.00	A
2019-06-02 23:00:00	0	0.00	A
2019-06-03 00:00:00	0	0.00	A
2019-06-03 01:00:00	0	0.00	A
2019-06-03 02:00:00	0	19.36	A

[5 rows x 57 columns]

```
[4]: from autogluon.tabular import TabularDataset, TabularPredictor
      from autogluon.timeseries import TimeSeriesDataFrame
```



```

import numpy as np
train_data = TabularDataset('X_train_raw.csv')
# set group column of train_data be increasing from 0 to 7 based on time, the
↳ first 1/8 of the data is group 0, the second 1/8 of the data is group 1, etc.
train_data['ds'] = pd.to_datetime(train_data['ds'])
train_data = train_data.sort_values(by='ds')

# # print size of the group for each location
# for loc in locations:
#     print(f"Location {loc}:")
#     print(train_data[train_data["location"] == loc].groupby('group').size())

# get end date of train data and subtract 3 months
split_time = pd.to_datetime(train_data["ds"]).max() - pd.
↳ Timedelta(hours=tune_and_test_length)
train_set = TabularDataset(train_data[train_data["ds"] < split_time])
test_set = TabularDataset(train_data[train_data["ds"] >= split_time])
if use_groups:
    test_set = test_set.drop(columns=['group'])

if do_drop_ds:
    train_set = train_set.drop(columns=['ds'])
    test_set = test_set.drop(columns=['ds'])
    train_data = train_data.drop(columns=['ds'])

def normalize_sample_weights_per_location(df):
    for loc in locations:
        loc_df = df[df["location"] == loc]
        loc_df["sample_weight"] = loc_df["sample_weight"] /
↳ loc_df["sample_weight"].sum() * loc_df.shape[0]
        df[df["location"] == loc] = loc_df
    return df

tuning_data = None
if use_tune_data:
    train_data = train_set
    if use_test_data:
        # split test_set in half, use first half for tuning
        tuning_data, test_data = [], []
        for loc in locations:
            loc_test_set = test_set[test_set["location"] == loc]
            loc_tuning_data = loc_test_set.iloc[:len(loc_test_set)//2]
            loc_test_data = loc_test_set.iloc[len(loc_test_set)//2:]
            tuning_data.append(loc_tuning_data)
            test_data.append(loc_test_data)

```

```

        tuning_data = pd.concat(tuning_data)
        test_data = pd.concat(test_data)
        print("Shapes of tuning and test", tuning_data.shape[0], test_data.
↪shape[0], tuning_data.shape[0] + test_data.shape[0])

    else:
        tuning_data = test_set
        print("Shape of tuning", tuning_data.shape[0])

        # ensure sample weights for your tuning data sum to the number of rows in
↪the tuning data.
        tuning_data = normalize_sample_weights_per_location(tuning_data)

else:
    if use_test_data:
        train_data = train_set
        test_data = test_set
        print("Shape of test", test_data.shape[0])

    # ensure sample weights for your training (or tuning) data sum to the number of
↪rows in the training (or tuning) data.
    train_data = normalize_sample_weights_per_location(train_data)
    if use_test_data:
        test_data = normalize_sample_weights_per_location(test_data)

```

Shape of test 5791

```

[5]: if run_analysis:
        import autogluon.eda.auto as auto
        auto.dataset_overview(train_data=train_data, test_data=test_data,
↪label="y", sample=None)

```

train_data dataset summary

	count	unique	top	freq	mean	\
absolute_humidity_2m:gm3	87160	757			6.138632	
air_density_2m:kgm3	87160	1370			1.253802	
ceiling_height_agl:m	72139	59833			2864.542561	
clear_sky_energy_1h:J	87160	45557			518223.98768	
clear_sky_energy_1h:J_not_shifted	87160	45557			518234.059562	
clear_sky_rad:W	87160	19511			143.951884	
cloud_base_agl:m	81279	61233			1736.160546	
dew_or_rime:idx	87160	9			0.009491	
dew_point_2m:K	87160	2001			275.537267	
diffuse_rad:W	87160	10980			39.210491	
diffuse_rad_1h:J	87160	45515			141509.744358	
diffuse_rad_1h:J_not_shifted	87160	45515			141510.381097	

direct_rad:W	87160	13914	50.139922
direct_rad_1h:J	87160	39280	180354.638612
direct_rad_1h:J_not_shifted	87160	39281	180365.163277
effective_cloud_cover:p	87160	5652	67.118857
elevation:m	87160	3	11.411014
fresh_snow_12h:cm	87160	121	0.096432
fresh_snow_12h:cm_not_shifted	87160	121	0.096392
fresh_snow_1h:cm	87160	39	0.008136
fresh_snow_1h:cm_not_shifted	87160	39	0.008132
fresh_snow_24h:cm	87160	158	0.191661
fresh_snow_24h:cm_not_shifted	87160	158	0.191621
fresh_snow_3h:cm	87160	68	0.024312
fresh_snow_3h:cm_not_shifted	87160	68	0.024302
fresh_snow_6h:cm	87160	94	0.048428
fresh_snow_6h:cm_not_shifted	87160	94	0.048398
is_day:idx	87160	5	0.482965
is_estimated	87232	2	0.05968
is_in_shadow:idx	87160	5	0.564895
location	87232	3	A 31924
msl_pressure:hPa	87160	3693	1009.291473
precip_5min:mm	87160	270	0.005788
precip_type_5min:idx	87160	15	0.084236
pressure_100m:hPa	87160	3705	995.621975
pressure_50m:hPa	87160	3758	1001.745166
prob_rime:p	87160	1723	0.697669
rain_water:kgm2	87160	39	0.010136
relative_humidity_1000hPa:p	87160	3787	73.860635
sample_weight	87232	1	1.0
sfc_pressure:hPa	87160	3780	1007.89561
snow_depth:cm	87160	483	0.197251
snow_melt_10min:mm	87160	63	0.000245
snow_water:kgm2	87160	161	0.09109
sun_azimuth:d	87160	82801	179.660078
sun_elevation:d	87160	71854	-1.225457
super_cooled_liquid_water:kgm2	87160	53	0.058341
t_1000hPa:K	87160	1986	279.712685
total_cloud_cover:p	87160	5546	73.819247
visibility:m	87160	85645	33233.674454
wind_speed_10m:ms	87160	594	3.025581
wind_speed_u_10m:ms	87160	988	0.664335
wind_speed_v_10m:ms	87160	848	0.694845
wind_speed_w_1000hPa:ms	87160	9	0.000002
y	87232	10750	287.954185

	std	min	25% \
absolute_humidity_2m:gm3	2.73761	0.5	4.1
air_density_2m:kgm3	0.036657	1.13925	1.22875
ceiling_height_agl:m	2531.428872	27.8	1085.2625

clear_sky_energy_1h:J	828407.396747	0.0	0.0
clear_sky_energy_1h:J_not_shifted	828409.158425	0.0	0.0
clear_sky_rad:W	230.149085	0.0	0.0
cloud_base_agl:m	1797.954658	27.8	598.2875
dew_or_rime:idx	0.234376	-1.0	0.0
dew_point_2m:K	6.846723	247.425	271.0
diffuse_rad:W	60.603659	0.0	0.0
diffuse_rad_1h:J	216223.834057	0.0	0.0
diffuse_rad_1h:J_not_shifted	216223.195823	0.0	0.0
direct_rad:W	113.07899	0.0	0.0
direct_rad_1h:J	402272.467762	0.0	0.0
direct_rad_1h:J_not_shifted	402279.826859	0.0	0.0
effective_cloud_cover:p	34.037938	0.0	42.41875
elevation:m	7.881548	6.0	6.0
fresh_snow_12h:cm	0.735619	0.0	0.0
fresh_snow_12h:cm_not_shifted	0.735566	0.0	0.0
fresh_snow_1h:cm	0.107502	0.0	0.0
fresh_snow_1h:cm_not_shifted	0.107497	0.0	0.0
fresh_snow_24h:cm	1.140022	0.0	0.0
fresh_snow_24h:cm_not_shifted	1.139991	0.0	0.0
fresh_snow_3h:cm	0.267721	0.0	0.0
fresh_snow_3h:cm_not_shifted	0.26771	0.0	0.0
fresh_snow_6h:cm	0.45772	0.0	0.0
fresh_snow_6h:cm_not_shifted	0.457678	0.0	0.0
is_day:idx	0.485944	0.0	0.0
is_estimated	0.236894	0.0	0.0
is_in_shadow:idx	0.483131	0.0	0.0
location			
msl_pressure:hPa	12.998509	944.375	1001.275
precip_5min:mm	0.029771	0.0	0.0
precip_type_5min:idx	0.325388	0.0	0.0
pressure_100m:hPa	12.924683	929.975	987.69995
pressure_50m:hPa	12.982562	935.75	993.75
prob_rime:p	5.08106	0.0	0.0
rain_water:kgm2	0.042332	0.0	0.0
relative_humidity_1000hPa:p	14.160229	19.575	64.425
sample_weight	0.0	1.0	1.0
sfc_pressure:hPa	13.042592	941.55	999.85
snow_depth:cm	1.284395	0.0	0.0
snow_melt_10min:mm	0.003958	0.0	0.0
snow_water:kgm2	0.240712	0.0	0.0
sun_azimuth:d	97.308971	6.983	94.72475
sun_elevation:d	24.168008	-49.932	-18.737563
super_cooled_liquid_water:kgm2	0.106882	0.0	0.0
t_1000hPa:K	6.559438	258.025	275.15
total_cloud_cover:p	33.768818	0.0	53.725
visibility:m	18089.724083	132.375	16688.61875
wind_speed_10m:ms	1.752114	0.025	1.65

wind_speed_u_10m:ms	2.779236	-7.225	-1.35	
wind_speed_v_10m:ms	1.881059	-8.4	-0.55	
wind_speed_w_1000hPa:ms	0.006041	-0.1	0.0	
y	766.111697	0.0	0.0	
	50%	75%	max	dtypes \
absolute_humidity_2m:gm3	5.6	8.0	17.35	float64
air_density_2m:kgm3	1.2525	1.27675	1.441	float64
ceiling_height_agl:m	1859.4751	3925.32485	12285.775	float64
clear_sky_energy_1h:J	4304.55	777057.95	3006697.2	float64
clear_sky_energy_1h:J_not_shifted	4309.0	777229.375	3006697.2	float64
clear_sky_rad:W	1.6	216.2	835.65	float64
cloud_base_agl:m	1178.425	2081.0375	11673.725	float64
dew_or_rime:idx	0.0	0.0	1.0	float64
dew_point_2m:K	275.4	280.8	293.625	float64
diffuse_rad:W	0.875	64.325	334.75	float64
diffuse_rad_1h:J	9533.25	232995.65	1182265.4	float64
diffuse_rad_1h:J_not_shifted	9534.75	232989.0	1182265.4	float64
direct_rad:W	0.0	28.925	683.4	float64
direct_rad_1h:J	0.0	111380.225	2445897.0	float64
direct_rad_1h:J_not_shifted	0.0	111410.2	2445897.0	float64
effective_cloud_cover:p	79.675	98.475	100.0	float64
elevation:m	7.0	24.0	24.0	float64
fresh_snow_12h:cm	0.0	0.0	37.4	float64
fresh_snow_12h:cm_not_shifted	0.0	0.0	37.4	float64
fresh_snow_1h:cm	0.0	0.0	7.1	float64
fresh_snow_1h:cm_not_shifted	0.0	0.0	7.1	float64
fresh_snow_24h:cm	0.0	0.0	37.4	float64
fresh_snow_24h:cm_not_shifted	0.0	0.0	37.4	float64
fresh_snow_3h:cm	0.0	0.0	20.6	float64
fresh_snow_3h:cm_not_shifted	0.0	0.0	20.6	float64
fresh_snow_6h:cm	0.0	0.0	34.0	float64
fresh_snow_6h:cm_not_shifted	0.0	0.0	34.0	float64
is_day:idx	0.25	1.0	1.0	float64
is_estimated	0.0	0.0	1.0	int64
is_in_shadow:idx	1.0	1.0	1.0	float64
location				object
msl_pressure:hPa	1010.275	1018.35	1044.1	float64
precip_5min:mm	0.0	0.0	0.6225	float64
precip_type_5min:idx	0.0	0.0	5.0	float64
pressure_100m:hPa	996.7	1004.7	1030.875	float64
pressure_50m:hPa	1002.8	1010.825	1037.25	float64
prob_rime:p	0.0	0.0	96.775	float64
rain_water:kgm2	0.0	0.0	1.1	float64
relative_humidity_1000hPa:p	76.2	85.25	100.0	float64
sample_weight	1.0	1.0	1.0	int64
sfc_pressure:hPa	1008.925	1017.0	1043.725	float64
snow_depth:cm	0.0	0.0	18.2	float64

snow_melt_10min:mm	0.0	0.0	0.18	float64
snow_water:kgm2	0.0	0.1	5.65	float64
sun_azimuth:d	180.007	264.513	348.48752	float64
sun_elevation:d	-0.8645	15.234063	49.94375	float64
super_cooled_liquid_water:kgm2	0.0	0.1	1.375	float64
t_1000hPa:K	279.075	284.25	303.25	float64
total_cloud_cover:p	92.85	99.9	100.0	float64
visibility:m	37320.0515	48663.7615	75489.33	float64
wind_speed_10m:ms	2.7	4.05	13.275	float64
wind_speed_u_10m:ms	0.3	2.475	11.2	float64
wind_speed_v_10m:ms	0.725	1.875	8.825	float64
wind_speed_w_1000hPa:ms	0.0	0.0	0.1	float64
y	0.0	176.4	5733.42	float64

	missing_count	missing_ratio	raw_type	\
absolute_humidity_2m:gm3	72	0.000825	float	
air_density_2m:kgm3	72	0.000825	float	
ceiling_height_agl:m	15093	0.173021	float	
clear_sky_energy_1h:J	72	0.000825	float	
clear_sky_energy_1h:J_not_shifted	72	0.000825	float	
clear_sky_rad:W	72	0.000825	float	
cloud_base_agl:m	5953	0.068243	float	
dew_or_rime:idx	72	0.000825	float	
dew_point_2m:K	72	0.000825	float	
diffuse_rad:W	72	0.000825	float	
diffuse_rad_1h:J	72	0.000825	float	
diffuse_rad_1h:J_not_shifted	72	0.000825	float	
direct_rad:W	72	0.000825	float	
direct_rad_1h:J	72	0.000825	float	
direct_rad_1h:J_not_shifted	72	0.000825	float	
effective_cloud_cover:p	72	0.000825	float	
elevation:m	72	0.000825	float	
fresh_snow_12h:cm	72	0.000825	float	
fresh_snow_12h:cm_not_shifted	72	0.000825	float	
fresh_snow_1h:cm	72	0.000825	float	
fresh_snow_1h:cm_not_shifted	72	0.000825	float	
fresh_snow_24h:cm	72	0.000825	float	
fresh_snow_24h:cm_not_shifted	72	0.000825	float	
fresh_snow_3h:cm	72	0.000825	float	
fresh_snow_3h:cm_not_shifted	72	0.000825	float	
fresh_snow_6h:cm	72	0.000825	float	
fresh_snow_6h:cm_not_shifted	72	0.000825	float	
is_day:idx	72	0.000825	float	
is_estimated				int
is_in_shadow:idx	72	0.000825	float	
location				object
msl_pressure:hPa	72	0.000825	float	
precip_5min:mm	72	0.000825	float	

precip_type_5min:idx	72	0.000825	float
pressure_100m:hPa	72	0.000825	float
pressure_50m:hPa	72	0.000825	float
prob_rime:p	72	0.000825	float
rain_water:kgm2	72	0.000825	float
relative_humidity_1000hPa:p	72	0.000825	float
sample_weight			int
sfc_pressure:hPa	72	0.000825	float
snow_depth:cm	72	0.000825	float
snow_melt_10min:mm	72	0.000825	float
snow_water:kgm2	72	0.000825	float
sun_azimuth:d	72	0.000825	float
sun_elevation:d	72	0.000825	float
super_cooled_liquid_water:kgm2	72	0.000825	float
t_1000hPa:K	72	0.000825	float
total_cloud_cover:p	72	0.000825	float
visibility:m	72	0.000825	float
wind_speed_10m:ms	72	0.000825	float
wind_speed_u_10m:ms	72	0.000825	float
wind_speed_v_10m:ms	72	0.000825	float
wind_speed_w_1000hPa:ms	72	0.000825	float
y			float

	variable_type	special_types
absolute_humidity_2m:gm3	numeric	
air_density_2m:kgm3	numeric	
ceiling_height_agl:m	numeric	
clear_sky_energy_1h:J	numeric	
clear_sky_energy_1h:J_not_shifted	numeric	
clear_sky_rad:W	numeric	
cloud_base_agl:m	numeric	
dew_or_rime:idx	category	
dew_point_2m:K	numeric	
diffuse_rad:W	numeric	
diffuse_rad_1h:J	numeric	
diffuse_rad_1h:J_not_shifted	numeric	
direct_rad:W	numeric	
direct_rad_1h:J	numeric	
direct_rad_1h:J_not_shifted	numeric	
effective_cloud_cover:p	numeric	
elevation:m	category	
fresh_snow_12h:cm	numeric	
fresh_snow_12h:cm_not_shifted	numeric	
fresh_snow_1h:cm	numeric	
fresh_snow_1h:cm_not_shifted	numeric	
fresh_snow_24h:cm	numeric	
fresh_snow_24h:cm_not_shifted	numeric	
fresh_snow_3h:cm	numeric	

fresh_snow_3h:cm_not_shifted	numeric
fresh_snow_6h:cm	numeric
fresh_snow_6h:cm_not_shifted	numeric
is_day:idx	category
is_estimated	category
is_in_shadow:idx	category
location	category
msl_pressure:hPa	numeric
precip_5min:mm	numeric
precip_type_5min:idx	category
pressure_100m:hPa	numeric
pressure_50m:hPa	numeric
prob_rime:p	numeric
rain_water:kgm2	numeric
relative_humidity_1000hPa:p	numeric
sample_weight	category
sfc_pressure:hPa	numeric
snow_depth:cm	numeric
snow_melt_10min:mm	numeric
snow_water:kgm2	numeric
sun_azimuth:d	numeric
sun_elevation:d	numeric
super_cooled_liquid_water:kgm2	numeric
t_1000hPa:K	numeric
total_cloud_cover:p	numeric
visibility:m	numeric
wind_speed_10m:ms	numeric
wind_speed_u_10m:ms	numeric
wind_speed_v_10m:ms	numeric
wind_speed_w_1000hPa:ms	category
y	numeric

test_data dataset summary

	count	unique	top	freq	mean	\
absolute_humidity_2m:gm3	5791	289			4.192639	
air_density_2m:kgm3	5791	640			1.280018	
ceiling_height_agl:m	4395	4247			3278.267059	
clear_sky_energy_1h:J	5788	3059			469132.824948	
clear_sky_energy_1h:J_not_shifted	5791	3059			468797.225142	
clear_sky_rad:W	5791	2046			130.246477	
cloud_base_agl:m	4934	4719			1733.271034	
dew_or_rime:idx	5791	9			-0.033716	
dew_point_2m:K	5791	948			270.733081	
diffuse_rad:W	5791	2237			42.175259	
diffuse_rad_1h:J	5788	3065			152461.828645	
diffuse_rad_1h:J_not_shifted	5791	3065			152259.007684	
direct_rad:W	5791	1829			51.829421	
direct_rad_1h:J	5788	2676			186526.762509	

direct_rad_1h:J_not_shifted	5791	2677	186384.946987
effective_cloud_cover:p	5791	2100	66.598541
elevation:m	5791	3	11.262131
fresh_snow_12h:cm	5788	82	0.415048
fresh_snow_12h:cm_not_shifted	5791	82	0.413935
fresh_snow_1h:cm	5788	23	0.032308
fresh_snow_1h:cm_not_shifted	5791	23	0.032171
fresh_snow_24h:cm	5788	108	0.808967
fresh_snow_24h:cm_not_shifted	5791	108	0.80594
fresh_snow_3h:cm	5788	42	0.100259
fresh_snow_3h:cm_not_shifted	5791	42	0.099724
fresh_snow_6h:cm	5788	60	0.204492
fresh_snow_6h:cm_not_shifted	5791	60	0.203626
is_day:idx	5791	5	0.488387
is_estimated	5791	1	1.0
is_in_shadow:idx	5791	5	0.555085
location	5791	3	A 2161
msl_pressure:hPa	5791	2040	1012.678587
precip_5min:mm	5791	63	0.003687
precip_type_5min:idx	5791	12	0.086039
pressure_100m:hPa	5791	2124	998.781639
pressure_50m:hPa	5791	2134	1005.02648
prob_rime:p	5791	350	1.502776
rain_water:kgm2	5791	7	0.000984
relative_humidity_1000hPa:p	5791	2051	70.810205
sample_weight	5791	1	1.0
sfc_pressure:hPa	5791	2148	1011.29959
snow_depth:cm	5791	78	0.131661
snow_melt_10min:mm	5791	38	0.000695
snow_water:kgm2	5791	68	0.078393
sun_azimuth:d	5791	5681	179.475343
sun_elevation:d	5791	5093	-0.927197
super_cooled_liquid_water:kgm2	5791	31	0.035175
t_1000hPa:K	5791	825	275.185991
total_cloud_cover:p	5791	1838	71.785616
visibility:m	5791	5784	29884.461577
wind_speed_10m:ms	5791	424	3.227599
wind_speed_u_10m:ms	5791	672	0.668019
wind_speed_v_10m:ms	5791	483	0.538344
wind_speed_w_1000hPa:ms	5791	8	-0.000155
y	5791	2304	272.991992

	std	min	25% \
absolute_humidity_2m:gm3	1.300644	1.1	3.35
air_density_2m:kgm3	0.024372	1.219	1.26375
ceiling_height_agl:m	2590.751931	27.925	1149.0625
clear_sky_energy_1h:J	689638.596662	0.0	0.0
clear_sky_energy_1h:J_not_shifted	689490.588724	0.0	0.0

clear_sky_rad:W	191.578221	0.0	0.0
cloud_base_agl:m	1987.046511	27.5	525.4375
dew_or_rime:idx	0.233147	-1.0	0.0
dew_point_2m:K	4.634046	255.05	268.33749
diffuse_rad:W	59.158733	0.0	0.0
diffuse_rad_1h:J	211011.771342	0.0	0.0
diffuse_rad_1h:J_not_shifted	210855.943527	0.0	0.0
direct_rad:W	110.450287	0.0	0.0
direct_rad_1h:J	393513.65175	0.0	0.0
direct_rad_1h:J_not_shifted	393435.26945	0.0	0.0
effective_cloud_cover:p	37.583548	0.0	33.6375
elevation:m	7.8114	6.0	6.0
fresh_snow_12h:cm	1.240733	0.0	0.0
fresh_snow_12h:cm_not_shifted	1.239777	0.0	0.0
fresh_snow_1h:cm	0.170919	0.0	0.0
fresh_snow_1h:cm_not_shifted	0.170652	0.0	0.0
fresh_snow_24h:cm	1.982565	0.0	0.0
fresh_snow_24h:cm_not_shifted	1.977015	0.0	0.0
fresh_snow_3h:cm	0.425766	0.0	0.0
fresh_snow_3h:cm_not_shifted	0.424781	0.0	0.0
fresh_snow_6h:cm	0.738932	0.0	0.0
fresh_snow_6h:cm_not_shifted	0.73781	0.0	0.0
is_day:idx	0.486436	0.0	0.0
is_estimated	0.0	1.0	1.0
is_in_shadow:idx	0.483636	0.0	0.0
location			
msl_pressure:hPa	13.953847	975.3	1003.875
precip_5min:mm	0.017701	0.0	0.0
precip_type_5min:idx	0.393918	0.0	0.0
pressure_100m:hPa	13.825369	962.4	989.9
pressure_50m:hPa	13.873049	968.45	996.087475
prob_rime:p	7.839203	0.0	0.0
rain_water:kgm2	0.009596	0.0	0.0
relative_humidity_1000hPa:p	14.940249	21.325	60.75
sample_weight	0.0	1.0	1.0
sfc_pressure:hPa	13.921629	974.55	1002.25
snow_depth:cm	0.635847	0.0	0.0
snow_melt_10min:mm	0.007333	0.0	0.0
snow_water:kgm2	0.189057	0.0	0.0
sun_azimuth:d	96.891969	14.913	94.264625
sun_elevation:d	20.775858	-44.28175	-17.109625
super_cooled_liquid_water:kgm2	0.084895	0.0	0.0
t_1000hPa:K	3.823552	261.975	272.8
total_cloud_cover:p	37.578218	0.0	41.8
visibility:m	14669.627165	1215.4	18727.05
wind_speed_10m:ms	1.869023	0.05	1.725
wind_speed_u_10m:ms	3.12501	-7.15	-1.75
wind_speed_v_10m:ms	1.838513	-5.3	-0.8

wind_speed_w_1000hPa:ms	0.005234	-0.1	0.0	
y	770.841016	-0.0	0.0	
	50%	75%	max	dtypes \
absolute_humidity_2m:gm3	4.3	5.05	7.7	float64
air_density_2m:kgm3	1.279	1.29375	1.37175	float64
ceiling_height_agl:m	2618.95	4661.025	12294.901	float64
clear_sky_energy_1h:J	11008.5	791394.0	2554290.5	float64
clear_sky_energy_1h:J_not_shifted	10525.3	789350.65	2554290.5	float64
clear_sky_rad:W	2.675	221.925	710.5	float64
cloud_base_agl:m	904.825	2014.962525	10674.3	float64
dew_or_rime:idx	0.0	0.0	1.0	float64
dew_point_2m:K	271.6	273.9	280.4	float64
diffuse_rad:W	1.775	78.4875	311.95	float64
diffuse_rad_1h:J	18860.9	279202.425	1071799.5	float64
diffuse_rad_1h:J_not_shifted	18852.8	278503.35	1071799.5	float64
direct_rad:W	0.0	34.0875	530.15	float64
direct_rad_1h:J	0.0	129529.5	1895533.0	float64
direct_rad_1h:J_not_shifted	0.0	128914.1	1895533.0	float64
effective_cloud_cover:p	85.375	99.975	100.0	float64
elevation:m	7.0	24.0	24.0	float64
fresh_snow_12h:cm	0.0	0.0	9.5	float64
fresh_snow_12h:cm_not_shifted	0.0	0.0	9.5	float64
fresh_snow_1h:cm	0.0	0.0	2.6	float64
fresh_snow_1h:cm_not_shifted	0.0	0.0	2.6	float64
fresh_snow_24h:cm	0.0	0.2	14.8	float64
fresh_snow_24h:cm_not_shifted	0.0	0.2	14.8	float64
fresh_snow_3h:cm	0.0	0.0	5.2	float64
fresh_snow_3h:cm_not_shifted	0.0	0.0	5.2	float64
fresh_snow_6h:cm	0.0	0.0	7.5	float64
fresh_snow_6h:cm_not_shifted	0.0	0.0	7.5	float64
is_day:idx	0.25	1.0	1.0	float64
is_estimated	1.0	1.0	1.0	int64
is_in_shadow:idx	1.0	1.0	1.0	float64
location				object
msl_pressure:hPa	1011.625	1023.8125	1041.3501	float64
precip_5min:mm	0.0	0.0	0.2475	float64
precip_type_5min:idx	0.0	0.0	3.0	float64
pressure_100m:hPa	997.9	1009.875	1028.05	float64
pressure_50m:hPa	1004.1	1016.1625	1034.45	float64
prob_rime:p	0.0	0.0	91.875	float64
rain_water:kgm2	0.0	0.0	0.175	float64
relative_humidity_1000hPa:p	73.1	82.075	98.0	float64
sample_weight	1.0	1.0	1.0	int64
sfc_pressure:hPa	1010.35	1022.5125	1040.8501	float64
snow_depth:cm	0.0	0.0	4.9	float64
snow_melt_10min:mm	0.0	0.0	0.14	float64
snow_water:kgm2	0.0	0.1	2.15	float64

sun_azimuth:d	179.52899	263.49875	347.37848	float64
sun_elevation:d	-0.79825	15.30325	41.13025	float64
super_cooled_liquid_water:kgm2	0.0	0.0	0.75	float64
t_1000hPa:K	275.175	277.525	285.1	float64
total_cloud_cover:p	96.65	100.0	100.0	float64
visibility:m	31311.025	40438.6635	66178.45	float64
wind_speed_10m:ms	2.9	4.45	10.2	float64
wind_speed_u_10m:ms	0.3	2.9	9.95	float64
wind_speed_v_10m:ms	0.625	1.825	7.15	float64
wind_speed_w_1000hPa:ms	0.0	0.0	0.1	float64
y	0.0	142.906699	5172.64	float64

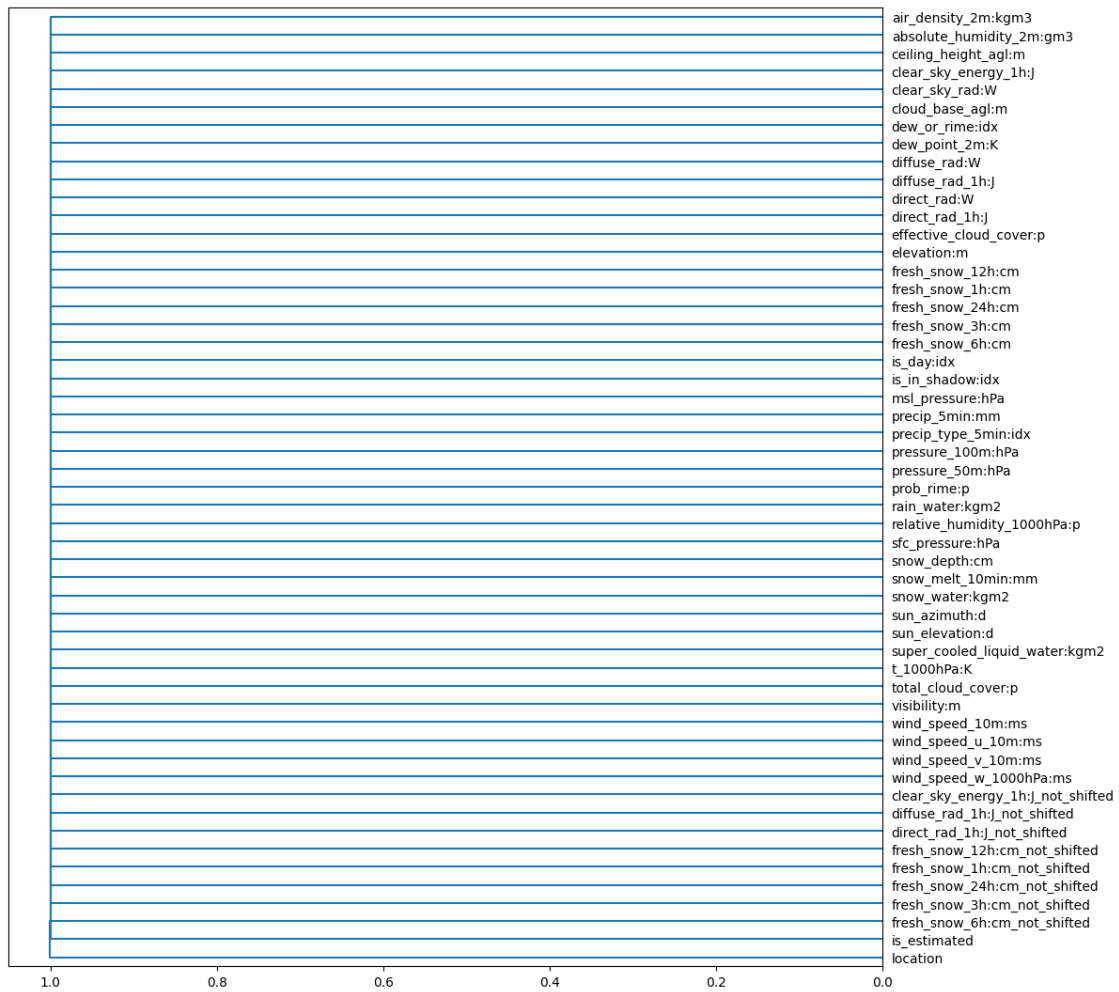
	missing_count	missing_ratio	raw_type	\
absolute_humidity_2m:gm3			float	
air_density_2m:kgm3			float	
ceiling_height_agl:m	1396	0.241064	float	
clear_sky_energy_1h:J	3	0.000518	float	
clear_sky_energy_1h:J_not_shifted			float	
clear_sky_rad:W			float	
cloud_base_agl:m	857	0.147988	float	
dew_or_rime:idx			float	
dew_point_2m:K			float	
diffuse_rad:W			float	
diffuse_rad_1h:J	3	0.000518	float	
diffuse_rad_1h:J_not_shifted			float	
direct_rad:W			float	
direct_rad_1h:J	3	0.000518	float	
direct_rad_1h:J_not_shifted			float	
effective_cloud_cover:p			float	
elevation:m			float	
fresh_snow_12h:cm	3	0.000518	float	
fresh_snow_12h:cm_not_shifted			float	
fresh_snow_1h:cm	3	0.000518	float	
fresh_snow_1h:cm_not_shifted			float	
fresh_snow_24h:cm	3	0.000518	float	
fresh_snow_24h:cm_not_shifted			float	
fresh_snow_3h:cm	3	0.000518	float	
fresh_snow_3h:cm_not_shifted			float	
fresh_snow_6h:cm	3	0.000518	float	
fresh_snow_6h:cm_not_shifted			float	
is_day:idx			float	
is_estimated			int	
is_in_shadow:idx			float	
location			object	
msl_pressure:hPa			float	
precip_5min:mm			float	
precip_type_5min:idx			float	
pressure_100m:hPa			float	

pressure_50m:hPa	float
prob_rime:p	float
rain_water:kgm2	float
relative_humidity_1000hPa:p	float
sample_weight	int
sfc_pressure:hPa	float
snow_depth:cm	float
snow_melt_10min:mm	float
snow_water:kgm2	float
sun_azimuth:d	float
sun_elevation:d	float
super_cooled_liquid_water:kgm2	float
t_1000hPa:K	float
total_cloud_cover:p	float
visibility:m	float
wind_speed_10m:ms	float
wind_speed_u_10m:ms	float
wind_speed_v_10m:ms	float
wind_speed_w_1000hPa:ms	float
y	float

	variable_type	special_types
absolute_humidity_2m:gm3	numeric	
air_density_2m:kgm3	numeric	
ceiling_height_agl:m	numeric	
clear_sky_energy_1h:J	numeric	
clear_sky_energy_1h:J_not_shifted	numeric	
clear_sky_rad:W	numeric	
cloud_base_agl:m	numeric	
dew_or_rime:idx	category	
dew_point_2m:K	numeric	
diffuse_rad:W	numeric	
diffuse_rad_1h:J	numeric	
diffuse_rad_1h:J_not_shifted	numeric	
direct_rad:W	numeric	
direct_rad_1h:J	numeric	
direct_rad_1h:J_not_shifted	numeric	
effective_cloud_cover:p	numeric	
elevation:m	category	
fresh_snow_12h:cm	numeric	
fresh_snow_12h:cm_not_shifted	numeric	
fresh_snow_1h:cm	numeric	
fresh_snow_1h:cm_not_shifted	numeric	
fresh_snow_24h:cm	numeric	
fresh_snow_24h:cm_not_shifted	numeric	
fresh_snow_3h:cm	numeric	
fresh_snow_3h:cm_not_shifted	numeric	
fresh_snow_6h:cm	numeric	

fresh_snow_6h:cm_not_shifted	numeric
is_day:idx	category
is_estimated	category
is_in_shadow:idx	category
location	category
msl_pressure:hPa	numeric
precip_5min:mm	numeric
precip_type_5min:idx	category
pressure_100m:hPa	numeric
pressure_50m:hPa	numeric
prob_rime:p	numeric
rain_water:kgm2	category
relative_humidity_1000hPa:p	numeric
sample_weight	category
sfc_pressure:hPa	numeric
snow_depth:cm	numeric
snow_melt_10min:mm	numeric
snow_water:kgm2	numeric
sun_azimuth:d	numeric
sun_elevation:d	numeric
super_cooled_liquid_water:kgm2	numeric
t_1000hPa:K	numeric
total_cloud_cover:p	numeric
visibility:m	numeric
wind_speed_10m:ms	numeric
wind_speed_u_10m:ms	numeric
wind_speed_v_10m:ms	numeric
wind_speed_w_1000hPa:ms	category
y	numeric

1.0.1 Feature Distance

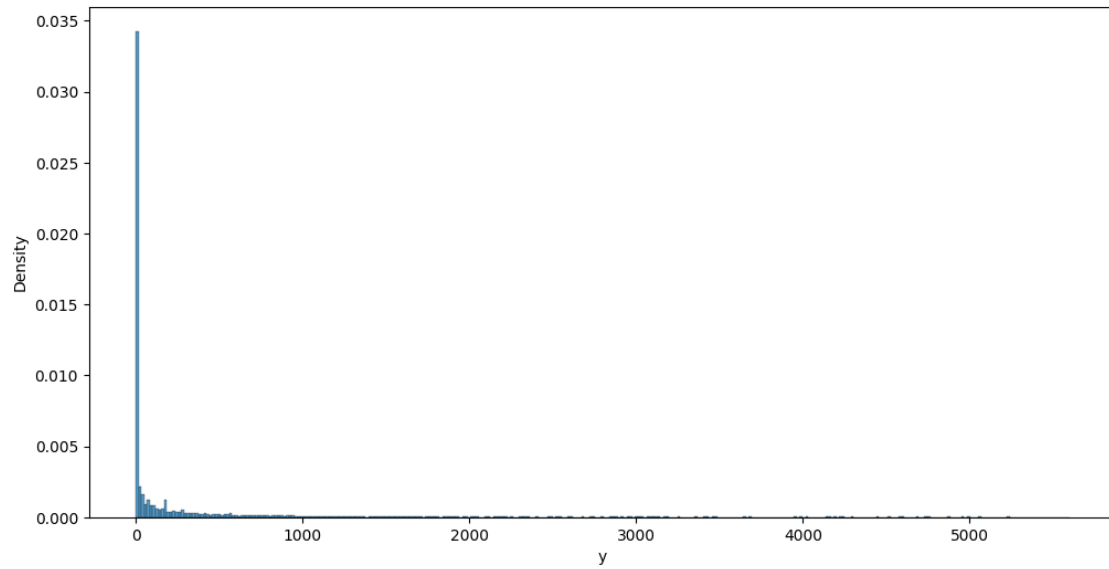


```
[6]: if run_analysis:
      auto.target_analysis(train_data=train_data, label="y")
```

1.1 Target variable analysis

	count	mean	std	min	25%	50%	75%	max	dtypes	\
y	10000	299.128516	787.495283	0.0	0.0	0.0	183.7125	5596.36	float64	

	unique	missing_count	missing_ratio	raw_type	special_types
y	2419			float	

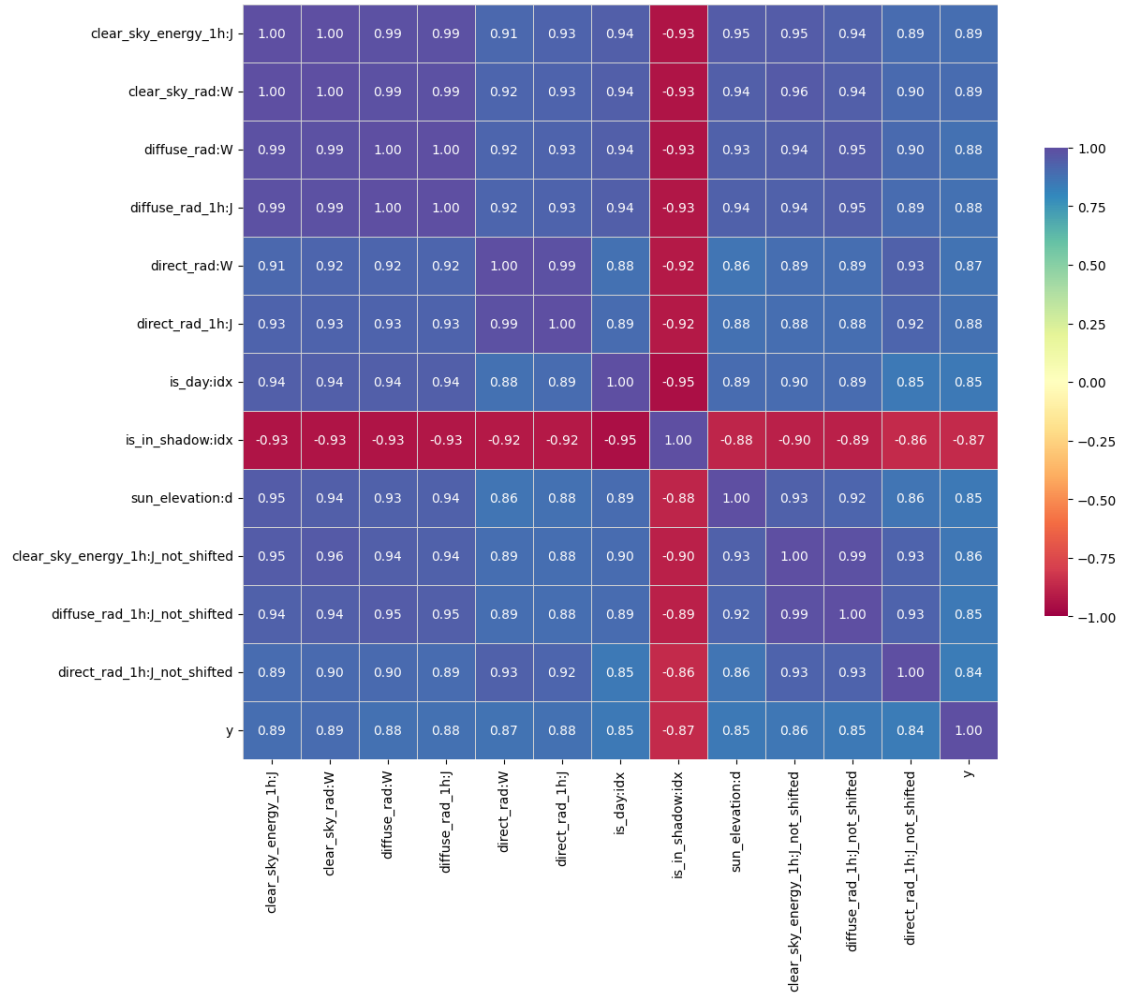


1.1.1 Distribution fits for target variable

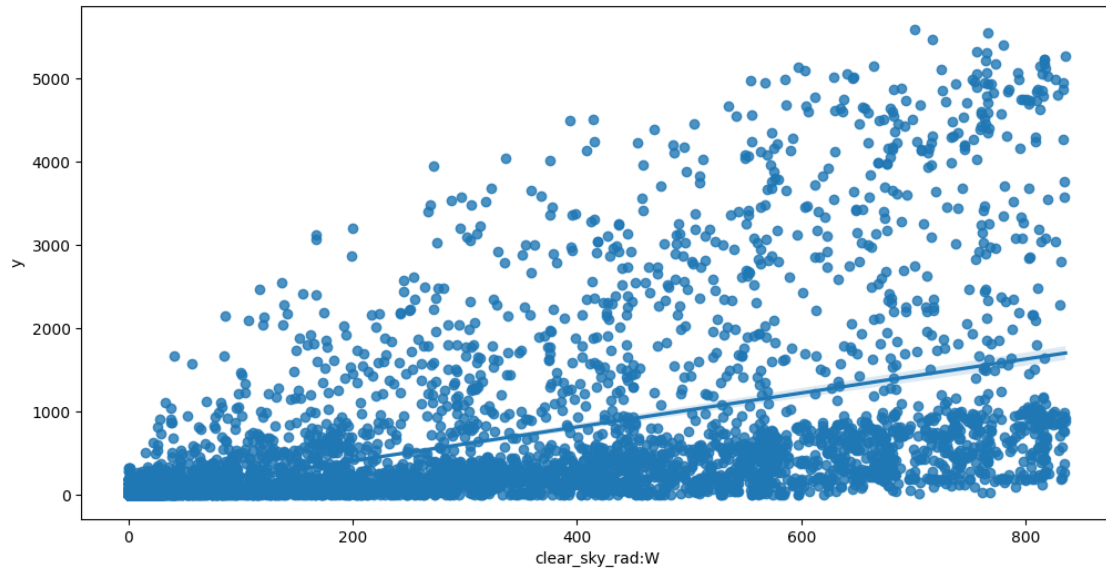
- none of the [attempted](#) distribution fits satisfy specified minimum p-value threshold: 0.01

1.1.2 Target variable correlations

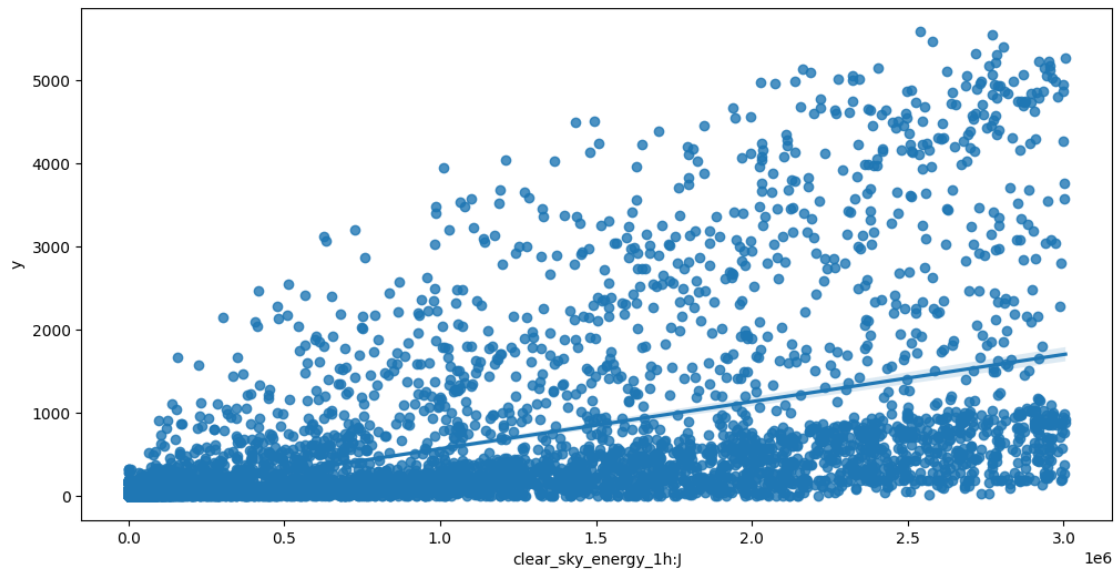
`train_data` - spearman correlation matrix; focus: absolute correlation for $y \geq 0.5$
(sample size: 10000)



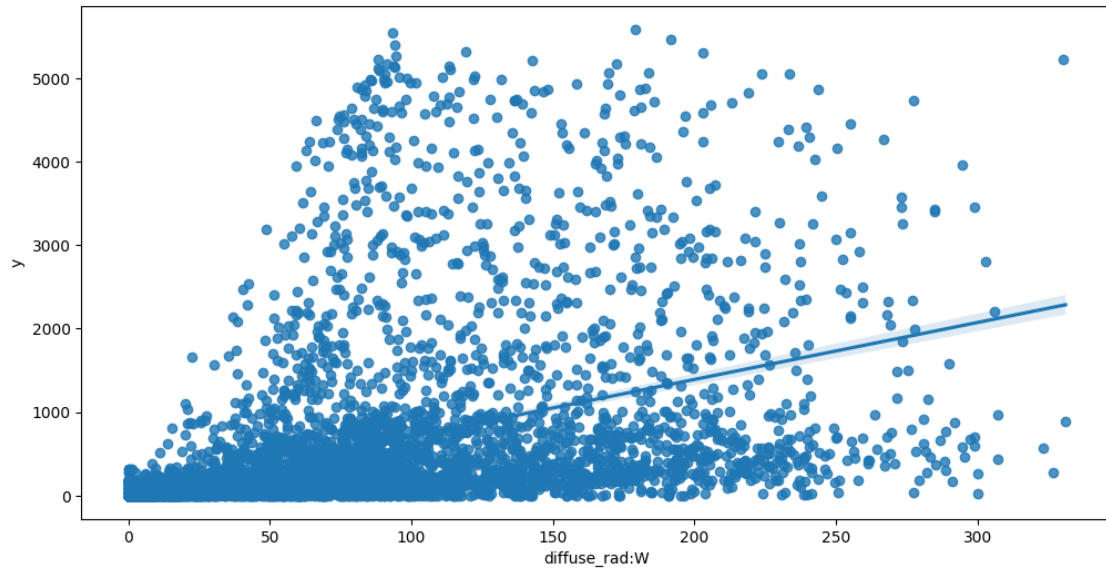
Feature interaction between clear_sky_rad:W/y in train_data (sample size: 10000)



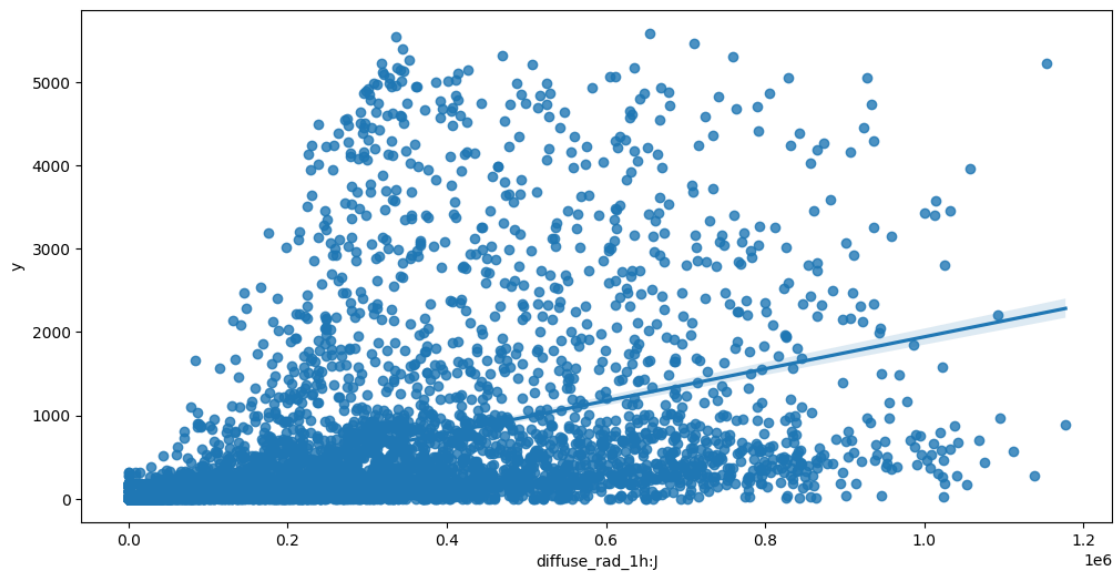
Feature interaction between clear_sky_energy_1h:J/y in train_data (sample size: 10000)



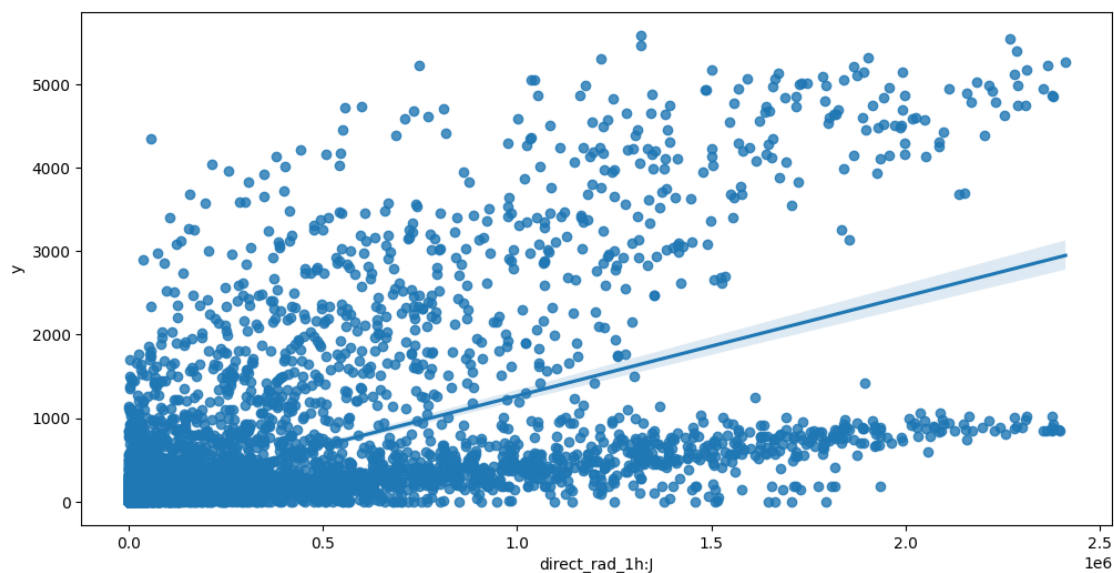
Feature interaction between diffuse_rad:W/y in train_data (sample size: 10000)



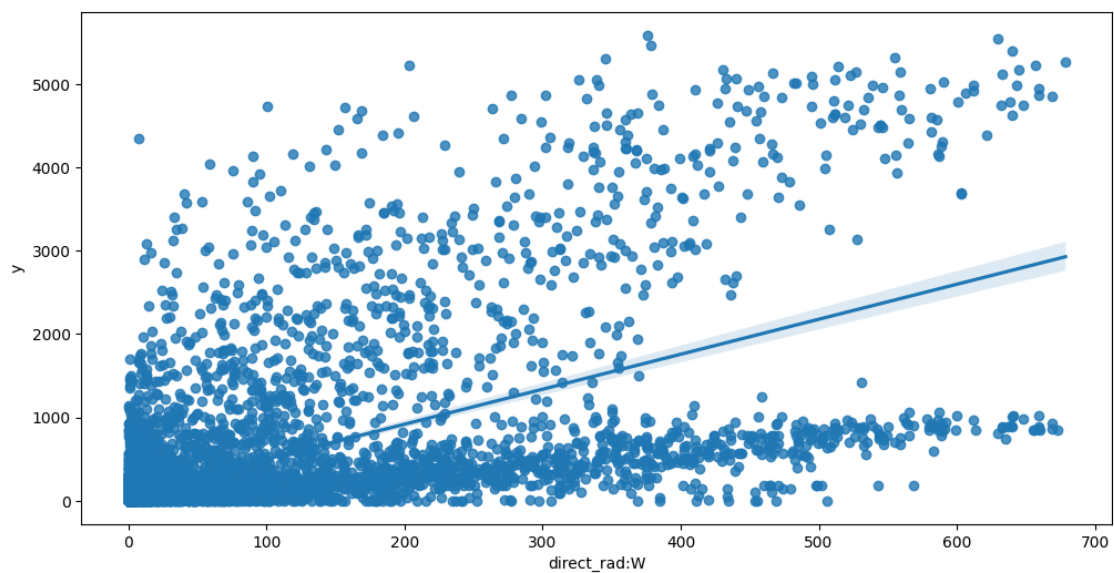
Feature interaction between diffuse_rad_1h:J/y in train_data (sample size: 10000)



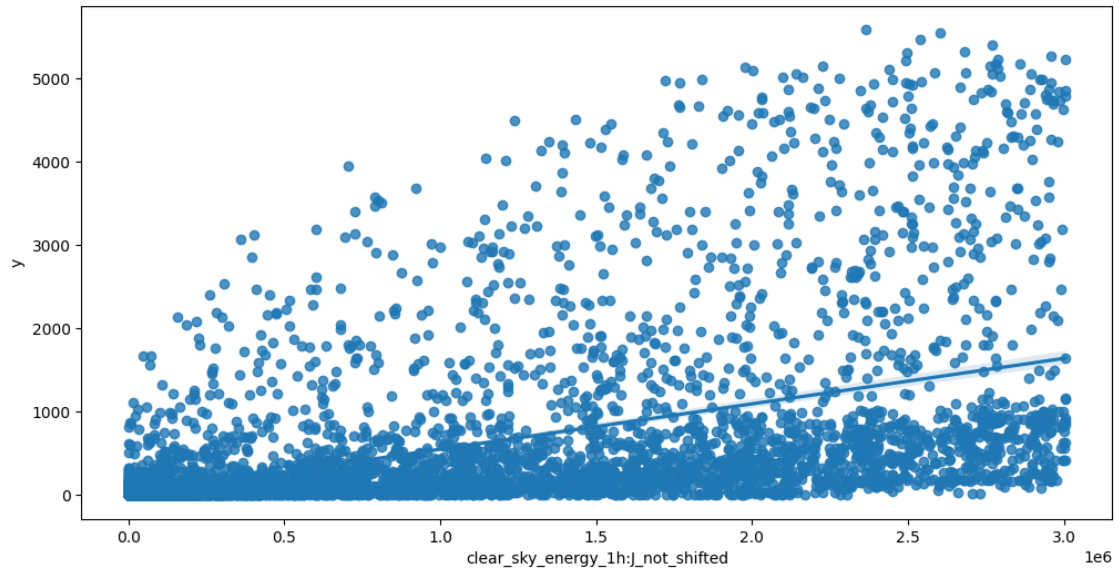
Feature interaction between direct_rad_1h:J/y in train_data (sample size: 10000)



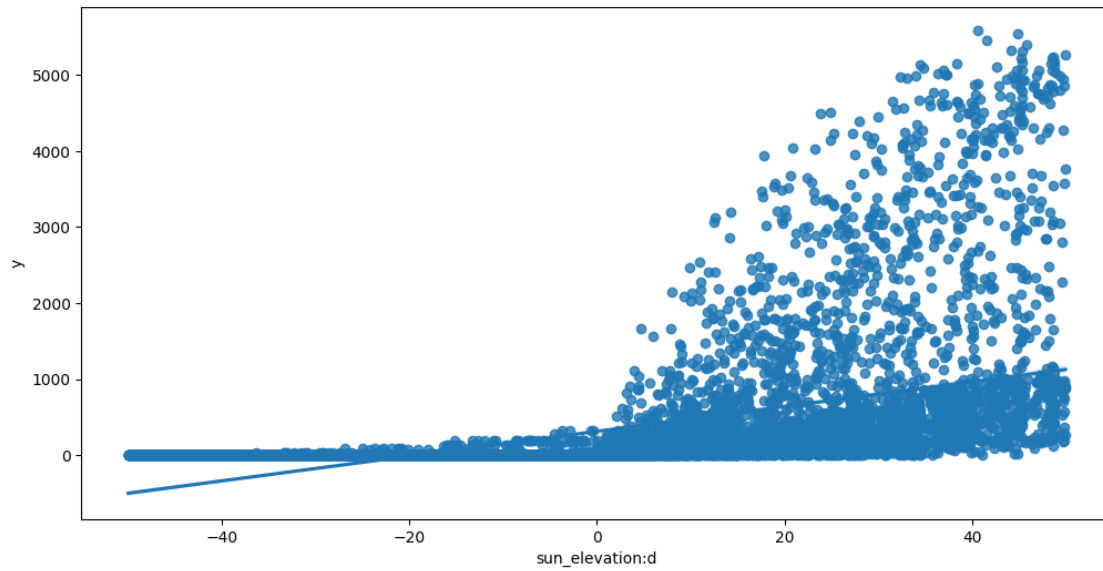
Feature interaction between direct_rad:W/y in train_data (sample size: 10000)



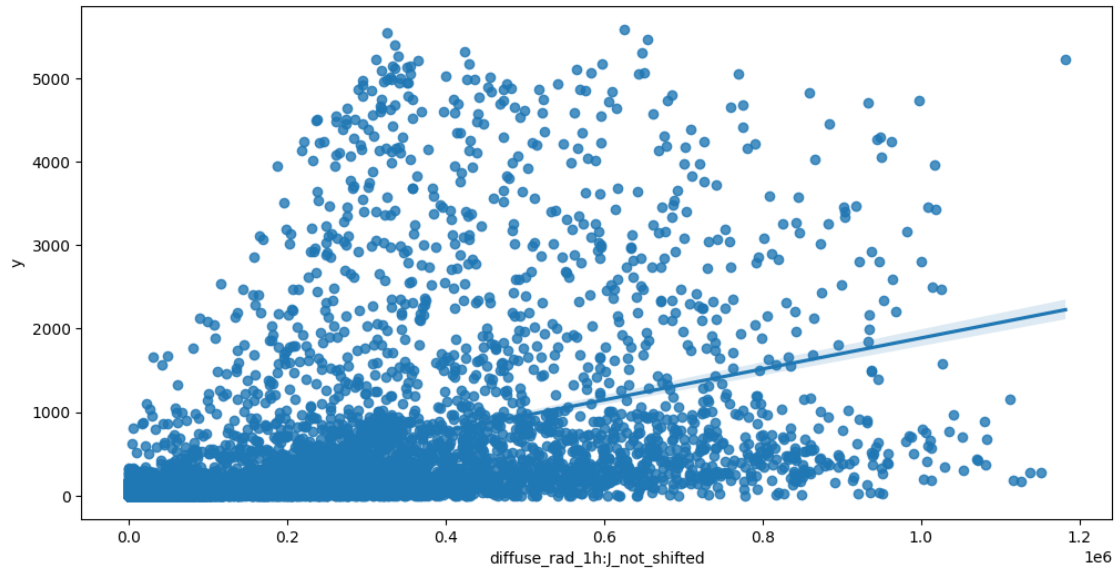
Feature interaction between clear_sky_energy_1h:J_not_shifted/y in train_data (sample size: 10000)



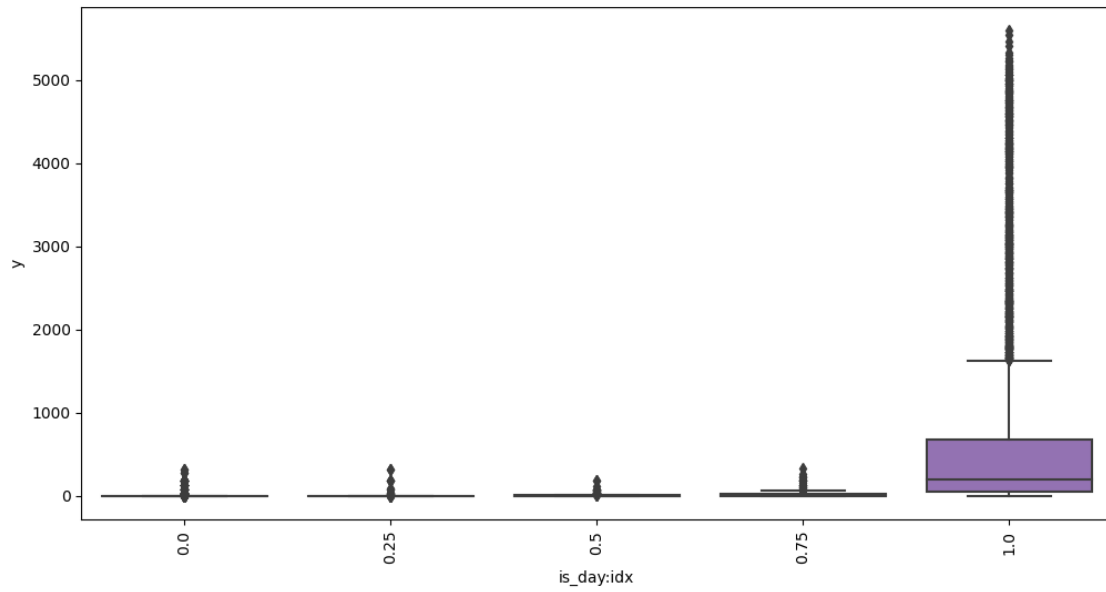
Feature interaction between `sun_elevation:d/y` in `train_data` (sample size: 10000)



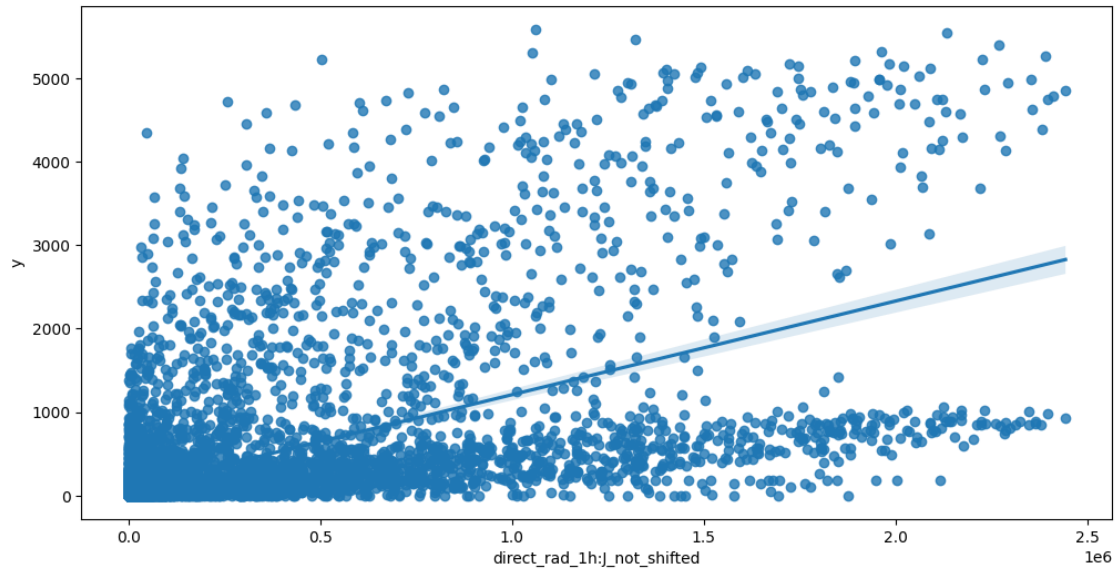
Feature interaction between `diffuse_rad_1h:J_not_shifted/y` in `train_data` (sample size: 10000)



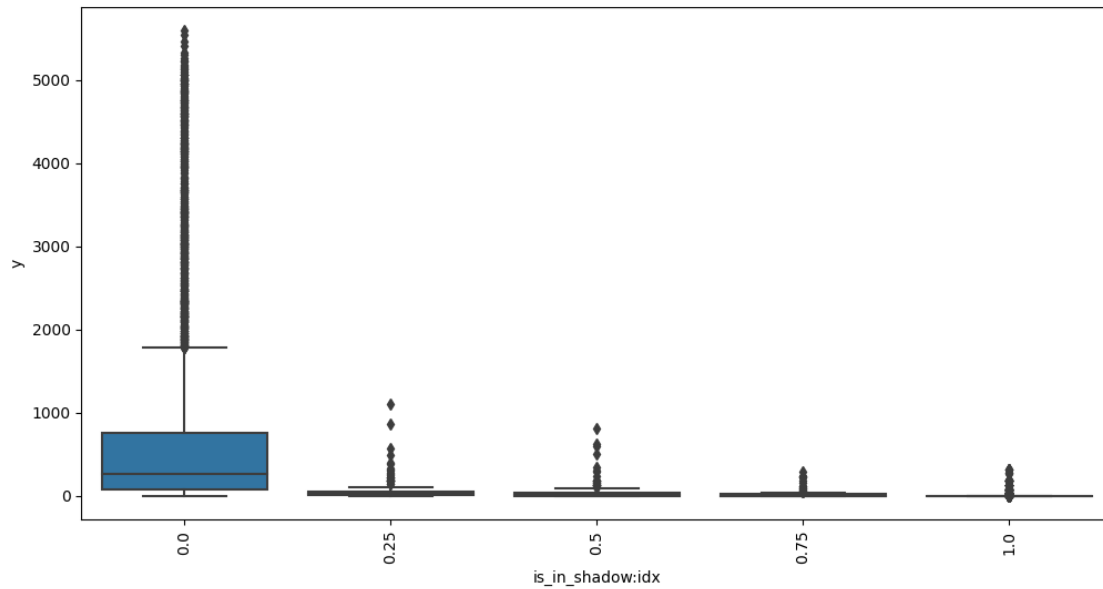
Feature interaction between `is_day:idx/y` in `train_data` (sample size: 10000)



Feature interaction between `direct_rad_1h:J_not_shifted/y` in `train_data` (sample size: 10000)



Feature interaction between `is_in_shadow:idx/y` in `train_data` (sample size: 10000)



2 Starting

```
[7]: import os
```

```

# Get the last submission number
last_submission_number = int(max([int(filename.split('_')[1].split('.')[0]) for
    ↪filename in os.listdir('submissions') if "submission" in filename]))
print("Last submission number:", last_submission_number)
print("Now creating submission number:", last_submission_number + 1)

# Create the new filename
new_filename = f'submission_{last_submission_number + 1}'

hello = os.environ.get('HELLO')
if hello is not None:
    new_filename += f'_{hello}'

print("New filename:", new_filename)

```

Last submission number: 88
 Now creating submission number: 89
 New filename: submission_89

```
[8]: predictors = [None, None, None]
```

```

[9]: def fit_predictor_for_location(loc):
    print(f"Training model for location {loc}...")
    # sum of sample weights for this location, and number of rows, for both
    ↪train and tune data and test data
    print("Train data sample weight sum:", train_data[train_data["location"] ==
    ↪loc]["sample_weight"].sum())
    print("Train data number of rows:", train_data[train_data["location"] ==
    ↪loc].shape[0])
    if use_tune_data:
        print("Tune data sample weight sum:",
    ↪tuning_data[tuning_data["location"] == loc]["sample_weight"].sum())
        print("Tune data number of rows:", tuning_data[tuning_data["location"]
    ↪== loc].shape[0])
    if use_test_data:
        print("Test data sample weight sum:", test_data[test_data["location"]
    ↪== loc]["sample_weight"].sum())
        print("Test data number of rows:", test_data[test_data["location"] ==
    ↪loc].shape[0])
    predictor = TabularPredictor(
        label=label,
        eval_metric=metric,
        path=f"AutogluonModels/{new_filename}_{loc}",
        sample_weight=sample_weight,
        weight_evaluation=weight_evaluation,
        groups="group" if use_groups else None,
    ).fit(

```



```

        train_data=train_data[train_data["location"] == loc],
        time_limit=time_limit,
        presets=presets,
        num_stack_levels=num_stack_levels,
        num_bag_folds=num_bag_folds if not use_groups else 2, # just put
        ↪somethin, will be overwritten anyways
        tuning_data=tuning_data[tuning_data["location"] == loc] if
        ↪use_tune_data else None,
        use_bag_holdout=use_bag_holdout,
        holdout_frac=holdout_frac,
    )

    # evaluate on test data
    if use_test_data:
        # drop sample_weight column
        t = test_data[test_data["location"] == loc]#.
        ↪drop(columns=["sample_weight"])
        perf = predictor.evaluate(t)
        print("Evaluation on test data:")
        print(perf[predictor.eval_metric.name])

    return predictor

loc = "A"
predictors[0] = fit_predictor_for_location(loc)

```

Warning: path already exists! This predictor may overwrite an existing predictor! path="AutogluonModels/submission_89_A"

Presets specified: ['best_quality']

Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8, num_bag_sets=20

Beginning AutoGluon training ... Time limit = 300s

AutoGluon will save models to "AutogluonModels/submission_89_A/"

AutoGluon Version: 0.8.2

Python Version: 3.10.12

Operating System: Linux

Platform Machine: x86_64

Platform Version: #1 SMP Debian 5.10.197-1 (2023-09-29)

Disk Space Avail: 307.40 GB / 315.93 GB (97.3%)

Train Data Rows: 31924

Train Data Columns: 54

Label Column: y

Preprocessing data ...

AutoGluon infers your prediction problem is: 'regression' (because dtype of label-column == float and many unique label-values observed).

Label info (max, min, mean, stddev): (5733.42, 0.0, 632.68576, 1165.32372)

If 'regression' is not the correct problem_type, please manually specify the problem_type parameter during predictor init (You may specify problem_type as one of: ['binary', 'multiclass', 'regression'])

Using Feature Generators to preprocess the data ...

Fitting AutoMLPipelineFeatureGenerator...

Available Memory: 132022.96 MB

Train Data (Original) Memory Usage: 15.39 MB (0.0% of available memory)

Inferring data type of each feature based on column values. Set feature_metadata_in to manually specify special dtypes of the features.

Stage 1 Generators:

Fitting AsTypeFeatureGenerator...

Note: Converting 2 features to boolean dtype as they only contain 2 unique values.

Stage 2 Generators:

Fitting FillNaFeatureGenerator...

Stage 3 Generators:

Training model for location A...

Train data sample weight sum: 31924

Train data number of rows: 31924

Test data sample weight sum: 2161

Test data number of rows: 2161

Fitting IdentityFeatureGenerator...

Stage 4 Generators:

Fitting DropUniqueFeatureGenerator...

Stage 5 Generators:

Fitting DropDuplicatesFeatureGenerator...

Useless Original Features (Count: 2): ['sample_weight', 'location']

These features carry no predictive signal and should be manually investigated.

This is typically a feature which has the same value for all rows.

These features do not need to be present at inference time.

Types of features in original data (raw dtype, special dtypes):

('float', []) : 51 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]

('int', []) : 1 | ['is_estimated']

Types of features in processed data (raw dtype, special dtypes):

('float', []) : 50 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]

('int', ['bool']) : 2 | ['elevation:m', 'is_estimated']

0.2s = Fit runtime

52 features in original data used to generate 52 features in processed data.

Train Data (Processed) Memory Usage: 12.83 MB (0.0% of available memory)

Data preprocessing and feature engineering runtime = 0.22s ...

AutoGluon will gauge predictive performance using evaluation metric:

```
'mean_absolute_error'
```

This metric's sign has been flipped to adhere to being higher_is_better. The metric score can be multiplied by -1 to get the metric value.

To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:

```
{
    'NN_TORCH': {},
    'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {}],
    'GBMLarge'],
    'CAT': {},
    'XGB': {},
    'FASTAI': {},
    'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
    'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
    'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}
```

AutoGluon will fit 2 stack levels (L1 to L2) ...

Fitting 11 L1 models ...

Fitting model: KNeighborsUnif_BAG_L1 ... Training model for up to 199.8s of the
299.77s of remaining time.

```
-219.613      = Validation score    (-mean_absolute_error)
0.05s        = Training    runtime
1.17s        = Validation runtime
```

Fitting model: KNeighborsDist_BAG_L1 ... Training model for up to 198.48s of the
298.45s of remaining time.

```
-219.6317     = Validation score    (-mean_absolute_error)
0.05s         = Training    runtime
0.46s         = Validation runtime
```

Fitting model: LightGBMXT_BAG_L1 ... Training model for up to 197.86s of the
297.83s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy

```
-155.6317     = Validation score    (-mean_absolute_error)
42.62s        = Training    runtime
17.07s        = Validation runtime
```

Fitting model: LightGBM_BAG_L1 ... Training model for up to 147.4s of the
247.38s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with

```

ParallelLocalFoldFittingStrategy
    -164.9935      = Validation score    (-mean_absolute_error)
    51.56s        = Training   runtime
    16.2s         = Validation runtime
Fitting model: RandomForestMSE_BAG_L1 ... Training model for up to 92.14s of the
192.12s of remaining time.
    -180.6119      = Validation score    (-mean_absolute_error)
    11.32s        = Training   runtime
    1.3s          = Validation runtime
Fitting model: CatBoost_BAG_L1 ... Training model for up to 77.44s of the
177.41s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -174.1327      = Validation score    (-mean_absolute_error)
    62.05s        = Training   runtime
    0.09s         = Validation runtime
Fitting model: ExtraTreesMSE_BAG_L1 ... Training model for up to 14.29s of the
114.26s of remaining time.
    -181.9104      = Validation score    (-mean_absolute_error)
    2.52s         = Training   runtime
    1.32s         = Validation runtime
Fitting model: NeuralNetFastAI_BAG_L1 ... Training model for up to 8.39s of the
108.37s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -216.5736      = Validation score    (-mean_absolute_error)
    7.69s         = Training   runtime
    0.73s         = Validation runtime
Completed 1/20 k-fold bagging repeats ...
Fitting model: WeightedEnsemble_L2 ... Training model for up to 299.78s of the
99.54s of remaining time.
    -155.0187      = Validation score    (-mean_absolute_error)
    0.6s          = Training   runtime
    0.0s          = Validation runtime
Fitting 9 L2 models ...
Fitting model: LightGBMXT_BAG_L2 ... Training model for up to 98.92s of the
98.9s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -156.8064      = Validation score    (-mean_absolute_error)
    2.84s         = Training   runtime
    0.17s         = Validation runtime
Fitting model: LightGBM_BAG_L2 ... Training model for up to 94.19s of the 94.18s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -155.1538      = Validation score    (-mean_absolute_error)
    2.38s         = Training   runtime

```

```

    0.08s      = Validation runtime
Fitting model: RandomForestMSE_BAG_L2 ... Training model for up to 90.45s of the
90.44s of remaining time.
    -153.7704      = Validation score      (-mean_absolute_error)
    15.51s      = Training runtime
    1.51s      = Validation runtime
Fitting model: CatBoost_BAG_L2 ... Training model for up to 71.38s of the 71.36s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -154.4355      = Validation score      (-mean_absolute_error)
    5.99s      = Training runtime
    0.04s      = Validation runtime
Fitting model: ExtraTreesMSE_BAG_L2 ... Training model for up to 64.12s of the
64.1s of remaining time.
    -152.7918      = Validation score      (-mean_absolute_error)
    2.82s      = Training runtime
    1.37s      = Validation runtime
Fitting model: NeuralNetFastAI_BAG_L2 ... Training model for up to 57.86s of the
57.85s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -153.6223      = Validation score      (-mean_absolute_error)
    41.82s      = Training runtime
    0.73s      = Validation runtime
Fitting model: XGBoost_BAG_L2 ... Training model for up to 14.73s of the 14.72s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -153.833      = Validation score      (-mean_absolute_error)
    2.96s      = Training runtime
    0.13s      = Validation runtime
Fitting model: NeuralNetTorch_BAG_L2 ... Training model for up to 10.41s of the
10.39s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -160.3937      = Validation score      (-mean_absolute_error)
    9.85s      = Training runtime
    0.67s      = Validation runtime
Completed 1/20 k-fold bagging repeats ...
Fitting model: WeightedEnsemble_L3 ... Training model for up to 299.78s of the
-0.85s of remaining time.
    -151.0503      = Validation score      (-mean_absolute_error)
    0.58s      = Training runtime
    0.0s      = Validation runtime
AutoGluon training complete, total runtime = 301.48s ... Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =

```

```

TabularPredictor.load("AutogluonModels/submission_89_A/")
Evaluation: mean_absolute_error on test data: -189.0870959744424
    Note: Scores are always higher_is_better. This metric score can be
multiplied by -1 to get the metric value.
Evaluations on test data:
{
    "mean_absolute_error": -189.0870959744424,
    "root_mean_squared_error": -424.2306549554531,
    "mean_squared_error": -179971.64860393267,
    "r2": 0.8694190903973649,
    "pearsonr": 0.934134287991857,
    "median_absolute_error": -4.977496242523193
}

Evaluation on test data:
-189.0870959744424

```

```

[10]: loc = "B"
      predictors[1] = fit_predictor_for_location(loc)

```

```

Warning: path already exists! This predictor may overwrite an existing
predictor! path="AutogluonModels/submission_89_B"
Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8,
num_bag_sets=20
Beginning AutoGluon training ... Time limit = 300s
AutoGluon will save models to "AutogluonModels/submission_89_B/"
AutoGluon Version: 0.8.2
Python Version: 3.10.12
Operating System: Linux
Platform Machine: x86_64
Platform Version: #1 SMP Debian 5.10.197-1 (2023-09-29)
Disk Space Avail: 307.44 GB / 315.93 GB (97.3%)
Train Data Rows: 30792
Train Data Columns: 54
Label Column: y
Preprocessing data ...
AutoGluon infers your prediction problem is: 'regression' (because dtype of
label-column == float and many unique label-values observed).
    Label info (max, min, mean, stddev): (1152.3, -0.0, 97.67477, 195.03642)
    If 'regression' is not the correct problem_type, please manually specify
the problem_type parameter during predictor init (You may specify problem_type
as one of: ['binary', 'multiclass', 'regression'])
Using Feature Generators to preprocess the data ...
Fitting AutoMLPipelineFeatureGenerator...
    Available Memory: 130628.99 MB
    Train Data (Original) Memory Usage: 14.84 MB (0.0% of available memory)
    Inferring data type of each feature based on column values. Set
feature_metadata_in to manually specify special dtypes of the features.

```

```

Stage 1 Generators:
    Fitting AsTypeFeatureGenerator...
        Note: Converting 2 features to boolean dtype as they
only contain 2 unique values.
Stage 2 Generators:
    Fitting FillNaFeatureGenerator...
Stage 3 Generators:
    Fitting IdentityFeatureGenerator...
Stage 4 Generators:

Training model for location B...
Train data sample weight sum: 30792
Train data number of rows: 30792
Test data sample weight sum: 2051
Test data number of rows: 2051

    Fitting DropUniqueFeatureGenerator...
Stage 5 Generators:
    Fitting DropDuplicatesFeatureGenerator...
Useless Original Features (Count: 2): ['sample_weight', 'location']
    These features carry no predictive signal and should be manually
investigated.
    This is typically a feature which has the same value for all
rows.

    These features do not need to be present at inference time.
Types of features in original data (raw dtype, special dtypes):
    ('float', []) : 51 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]
    ('int', [])   : 1 | ['is_estimated']
Types of features in processed data (raw dtype, special dtypes):
    ('float', []) : 50 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]
    ('int', ['bool']) : 2 | ['elevation:m', 'is_estimated']
0.2s = Fit runtime
52 features in original data used to generate 52 features in processed
data.

Train Data (Processed) Memory Usage: 12.38 MB (0.0% of available memory)
Data preprocessing and feature engineering runtime = 0.28s ...
AutoGluon will gauge predictive performance using evaluation metric:
'mean_absolute_error'

    This metric's sign has been flipped to adhere to being higher_is_better.
The metric score can be multiplied by -1 to get the metric value.
    To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:
{
    'NN_TORCH': {},
    'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {}],

```

```

'GBMLarge'],
  'CAT': {},
  'XGB': {},
  'FASTAI': {},
  'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
  'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
  'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}

```

AutoGluon will fit 2 stack levels (L1 to L2) ...

Fitting 11 L1 models ...

Fitting model: KNeighborsUnif_BAG_L1 ... Training model for up to 199.76s of the 299.72s of remaining time.

-47.1734 = Validation score (-mean_absolute_error)

0.05s = Training runtime

0.47s = Validation runtime

Fitting model: KNeighborsDist_BAG_L1 ... Training model for up to 199.18s of the 299.13s of remaining time.

-47.0004 = Validation score (-mean_absolute_error)

0.04s = Training runtime

0.45s = Validation runtime

Fitting model: LightGBMXT_BAG_L1 ... Training model for up to 198.63s of the 298.59s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with

ParallelLocalFoldFittingStrategy

-28.7015 = Validation score (-mean_absolute_error)

45.34s = Training runtime

19.33s = Validation runtime

Fitting model: LightGBM_BAG_L1 ... Training model for up to 149.22s of the 249.18s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with

ParallelLocalFoldFittingStrategy

-30.3689 = Validation score (-mean_absolute_error)

51.94s = Training runtime

19.44s = Validation runtime

Fitting model: RandomForestMSE_BAG_L1 ... Training model for up to 93.45s of the 193.41s of remaining time.

-34.9439 = Validation score (-mean_absolute_error)

12.78s = Training runtime

1.27s = Validation runtime

Fitting model: CatBoost_BAG_L1 ... Training model for up to 78.86s of the 178.81s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with ParallelLocalFoldFittingStrategy

-33.6817 = Validation score (-mean_absolute_error)
63.78s = Training runtime
0.07s = Validation runtime

Fitting model: ExtraTreesMSE_BAG_L1 ... Training model for up to 13.89s of the 113.85s of remaining time.

-36.0229 = Validation score (-mean_absolute_error)
2.39s = Training runtime
1.26s = Validation runtime

Fitting model: NeuralNetFastAI_BAG_L1 ... Training model for up to 9.67s of the 109.62s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with ParallelLocalFoldFittingStrategy

-43.6085 = Validation score (-mean_absolute_error)
9.54s = Training runtime
0.76s = Validation runtime

Completed 1/20 k-fold bagging repeats ...

Fitting model: WeightedEnsemble_L2 ... Training model for up to 299.72s of the 98.83s of remaining time.

-28.5421 = Validation score (-mean_absolute_error)
0.59s = Training runtime
0.0s = Validation runtime

Fitting 9 L2 models ...

Fitting model: LightGBMXT_BAG_L2 ... Training model for up to 98.23s of the 98.21s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with ParallelLocalFoldFittingStrategy

-26.1364 = Validation score (-mean_absolute_error)
4.99s = Training runtime
0.35s = Validation runtime

Fitting model: LightGBM_BAG_L2 ... Training model for up to 91.89s of the 91.87s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with ParallelLocalFoldFittingStrategy

-25.7381 = Validation score (-mean_absolute_error)
3.31s = Training runtime
0.11s = Validation runtime

Fitting model: RandomForestMSE_BAG_L2 ... Training model for up to 87.26s of the 87.25s of remaining time.

-24.5054 = Validation score (-mean_absolute_error)
15.17s = Training runtime
1.32s = Validation runtime

Fitting model: CatBoost_BAG_L2 ... Training model for up to 70.23s of the 70.22s of remaining time.

Fitting 8 child models (S1F1 - S1F8) | Fitting with

```

ParallelLocalFoldFittingStrategy
    -25.7772      = Validation score    (-mean_absolute_error)
    10.57s       = Training   runtime
    0.04s        = Validation runtime
Fitting model: ExtraTreesMSE_BAG_L2 ... Training model for up to 58.47s of the
58.46s of remaining time.
    -24.4516      = Validation score    (-mean_absolute_error)
    2.63s         = Training   runtime
    1.42s         = Validation runtime
Fitting model: NeuralNetFastAI_BAG_L2 ... Training model for up to 53.85s of the
53.84s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -24.9852      = Validation score    (-mean_absolute_error)
    40.3s         = Training   runtime
    0.73s         = Validation runtime
Fitting model: XGBoost_BAG_L2 ... Training model for up to 12.18s of the 12.17s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -25.3105      = Validation score    (-mean_absolute_error)
    3.52s         = Training   runtime
    0.14s         = Validation runtime
Fitting model: NeuralNetTorch_BAG_L2 ... Training model for up to 7.2s of the
7.19s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -31.2335      = Validation score    (-mean_absolute_error)
    7.33s         = Training   runtime
    0.68s         = Validation runtime
Completed 1/20 k-fold bagging repeats ...
Fitting model: WeightedEnsemble_L3 ... Training model for up to 299.72s of the
-1.57s of remaining time.
    -24.2899      = Validation score    (-mean_absolute_error)
    0.6s          = Training   runtime
    0.0s          = Validation runtime
AutoGluon training complete, total runtime = 302.23s ... Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =
TabularPredictor.load("AutogluonModels/submission_89_B/")
Evaluation: mean_absolute_error on test data: -36.70692390951731
    Note: Scores are always higher_is_better. This metric score can be
multiplied by -1 to get the metric value.
Evaluations on test data:
{
    "mean_absolute_error": -36.70692390951731,
    "root_mean_squared_error": -88.26733573190592,
    "mean_squared_error": -7791.122557208996,

```

```

    "r2": 0.749414393838681,
    "pearsonr": 0.8834082340709369,
    "median_absolute_error": -5.018582344055176
}

```

Evaluation on test data:
-36.70692390951731

```

[11]: loc = "C"
      predictors[2] = fit_predictor_for_location(loc)

```

```

Presets specified: ['best_quality']
Stack configuration (auto_stack=True): num_stack_levels=1, num_bag_folds=8,
num_bag_sets=20
Beginning AutoGluon training ... Time limit = 300s
AutoGluon will save models to "AutogluonModels/submission_89_C/"
AutoGluon Version: 0.8.2
Python Version: 3.10.12
Operating System: Linux
Platform Machine: x86_64
Platform Version: #1 SMP Debian 5.10.197-1 (2023-09-29)
Disk Space Avail: 305.77 GB / 315.93 GB (96.8%)
Train Data Rows: 24516
Train Data Columns: 54
Label Column: y
Preprocessing data ...
AutoGluon infers your prediction problem is: 'regression' (because dtype of
label-column == float and label-values can't be converted to int).
    Label info (max, min, mean, stddev): (999.6, 0.0, 78.04527, 167.43618)
    If 'regression' is not the correct problem_type, please manually specify
the problem_type parameter during predictor init (You may specify problem_type
as one of: ['binary', 'multiclass', 'regression'])
Using Feature Generators to preprocess the data ...
Fitting AutoMLPipelineFeatureGenerator...
    Available Memory: 129143.0 MB
    Train Data (Original) Memory Usage: 11.82 MB (0.0% of available memory)
    Inferring data type of each feature based on column values. Set
feature_metadata_in to manually specify special dtypes of the features.
    Stage 1 Generators:
        Fitting AsTypeFeatureGenerator...
            Note: Converting 2 features to boolean dtype as they
only contain 2 unique values.
    Stage 2 Generators:
        Fitting FillNaFeatureGenerator...
    Stage 3 Generators:
        Fitting IdentityFeatureGenerator...
    Stage 4 Generators:
        Fitting DropUniqueFeatureGenerator...

```

```

Training model for location C...
Train data sample weight sum: 24516
Train data number of rows: 24516
Test data sample weight sum: 1579
Test data number of rows: 1579

    Stage 5 Generators:
        Fitting DropDuplicatesFeatureGenerator...
    Useless Original Features (Count: 2): ['sample_weight', 'location']
        These features carry no predictive signal and should be manually
investigated.

        This is typically a feature which has the same value for all
rows.

        These features do not need to be present at inference time.
    Types of features in original data (raw dtype, special dtypes):
        ('float', []) : 51 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]
        ('int', [])   : 1 | ['is_estimated']
    Types of features in processed data (raw dtype, special dtypes):
        ('float', []) : 50 | ['absolute_humidity_2m:gm3',
'air_density_2m:kgm3', 'ceiling_height_agl:m', 'clear_sky_energy_1h:J',
'clear_sky_rad:W', ...]
        ('int', ['bool']) : 2 | ['elevation:m', 'is_estimated']
    0.2s = Fit runtime
    52 features in original data used to generate 52 features in processed
data.

    Train Data (Processed) Memory Usage: 9.86 MB (0.0% of available memory)
Data preprocessing and feature engineering runtime = 0.26s ...
AutoGluon will gauge predictive performance using evaluation metric:
'mean_absolute_error'

    This metric's sign has been flipped to adhere to being higher_is_better.
The metric score can be multiplied by -1 to get the metric value.

    To change this, specify the eval_metric parameter of Predictor()
User-specified model hyperparameters to be fit:
{
    'NN_TORCH': {},
    'GBM': [{'extra_trees': True, 'ag_args': {'name_suffix': 'XT'}}, {}],
'GBMLarge'],
    'CAT': {},
    'XGB': {},
    'FASTAI': {},
    'RF': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',
'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
    'XT': [{'criterion': 'gini', 'ag_args': {'name_suffix': 'Gini',

```

```

'problem_types': ['binary', 'multiclass']}}, {'criterion': 'entropy', 'ag_args':
{'name_suffix': 'Entr', 'problem_types': ['binary', 'multiclass']}},
{'criterion': 'squared_error', 'ag_args': {'name_suffix': 'MSE',
'problem_types': ['regression', 'quantile']}}],
    'KNN': [{'weights': 'uniform', 'ag_args': {'name_suffix': 'Unif'}},
{'weights': 'distance', 'ag_args': {'name_suffix': 'Dist'}}],
}
AutoGluon will fit 2 stack levels (L1 to L2) ...
Fitting 11 L1 models ...
Fitting model: KNeighborsUnif_BAG_L1 ... Training model for up to 199.78s of the
299.74s of remaining time.
    -25.9293          = Validation score    (-mean_absolute_error)
    0.04s           = Training    runtime
    0.32s           = Validation runtime
Fitting model: KNeighborsDist_BAG_L1 ... Training model for up to 199.38s of the
299.34s of remaining time.
    -25.879          = Validation score    (-mean_absolute_error)
    0.04s           = Training    runtime
    0.31s           = Validation runtime
Fitting model: LightGBMXT_BAG_L1 ... Training model for up to 198.98s of the
298.95s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -16.9668          = Validation score    (-mean_absolute_error)
    40.0s            = Training    runtime
    9.99s            = Validation runtime
Fitting model: LightGBM_BAG_L1 ... Training model for up to 156.22s of the
256.18s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -18.1061          = Validation score    (-mean_absolute_error)
    50.33s           = Training    runtime
    9.92s            = Validation runtime
Fitting model: RandomForestMSE_BAG_L1 ... Training model for up to 103.18s of
the 203.15s of remaining time.
    -19.9705          = Validation score    (-mean_absolute_error)
    7.19s            = Training    runtime
    0.96s            = Validation runtime
Fitting model: CatBoost_BAG_L1 ... Training model for up to 94.69s of the
194.65s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -18.9437          = Validation score    (-mean_absolute_error)
    76.46s           = Training    runtime
    0.07s            = Validation runtime
Fitting model: ExtraTreesMSE_BAG_L1 ... Training model for up to 17.04s of the
117.0s of remaining time.
    -20.0336          = Validation score    (-mean_absolute_error)

```

```

1.6s      = Training  runtime
0.94s     = Validation runtime
Fitting model: NeuralNetFastAI_BAG_L1 ... Training model for up to 14.12s of the
114.08s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -22.0753      = Validation score    (-mean_absolute_error)
    13.15s       = Training  runtime
    0.58s        = Validation runtime
Completed 1/20 k-fold bagging repeats ...
Fitting model: WeightedEnsemble_L2 ... Training model for up to 299.74s of the
99.63s of remaining time.
    -16.9058      = Validation score    (-mean_absolute_error)
    0.53s        = Training  runtime
    0.0s         = Validation runtime
Fitting 9 L2 models ...
Fitting model: LightGBMXT_BAG_L2 ... Training model for up to 99.09s of the
99.06s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -17.4626      = Validation score    (-mean_absolute_error)
    4.1s         = Training  runtime
    0.19s        = Validation runtime
Fitting model: LightGBM_BAG_L2 ... Training model for up to 93.71s of the 93.69s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -17.171       = Validation score    (-mean_absolute_error)
    2.45s        = Training  runtime
    0.07s        = Validation runtime
Fitting model: RandomForestMSE_BAG_L2 ... Training model for up to 90.04s of the
90.02s of remaining time.
    -16.909       = Validation score    (-mean_absolute_error)
    9.77s        = Training  runtime
    0.89s        = Validation runtime
Fitting model: CatBoost_BAG_L2 ... Training model for up to 79.05s of the 79.04s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -17.2503      = Validation score    (-mean_absolute_error)
    7.13s        = Training  runtime
    0.04s        = Validation runtime
Fitting model: ExtraTreesMSE_BAG_L2 ... Training model for up to 70.72s of the
70.71s of remaining time.
    -16.7678      = Validation score    (-mean_absolute_error)
    1.7s         = Training  runtime
    1.0s         = Validation runtime
Fitting model: NeuralNetFastAI_BAG_L2 ... Training model for up to 67.65s of the

```

```

67.64s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -16.9428      = Validation score    (-mean_absolute_error)
    31.92s       = Training    runtime
    0.63s        = Validation runtime
Fitting model: XGBoost_BAG_L2 ... Training model for up to 34.45s of the 34.44s
of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -17.0161      = Validation score    (-mean_absolute_error)
    3.34s         = Training    runtime
    0.1s          = Validation runtime
Fitting model: NeuralNetTorch_BAG_L2 ... Training model for up to 29.85s of the
29.84s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -17.214       = Validation score    (-mean_absolute_error)
    25.68s        = Training    runtime
    0.56s         = Validation runtime
Fitting model: LightGBMLarge_BAG_L2 ... Training model for up to 2.93s of the
2.92s of remaining time.
    Fitting 8 child models (S1F1 - S1F8) | Fitting with
ParallelLocalFoldFittingStrategy
    -18.602       = Validation score    (-mean_absolute_error)
    3.2s          = Training    runtime
    0.08s         = Validation runtime
Completed 1/20 k-fold bagging repeats ...
Fitting model: WeightedEnsemble_L3 ... Training model for up to 299.74s of the
-1.61s of remaining time.
    -16.5349      = Validation score    (-mean_absolute_error)
    0.58s         = Training    runtime
    0.0s          = Validation runtime
AutoGluon training complete, total runtime = 302.22s ... Best model:
"WeightedEnsemble_L3"
TabularPredictor saved. To load, use: predictor =
TabularPredictor.load("AutogluonModels/submission_89_C/")
Evaluation: mean_absolute_error on test data: -33.45657412906985
    Note: Scores are always higher_is_better. This metric score can be
multiplied by -1 to get the metric value.
Evaluations on test data:
{
    "mean_absolute_error": -33.45657412906985,
    "root_mean_squared_error": -69.1526037399097,
    "mean_squared_error": -4782.082604008972,
    "r2": 0.749274551565037,
    "pearsonr": 0.8849729932242641,
    "median_absolute_error": -2.6433258056640625
}

```

```
}
```

Evaluation on test data:

-33.45657412906985

3 Submit

```
[12]: import pandas as pd
import matplotlib.pyplot as plt

train_data_with_dates = TabularDataset('X_train_raw.csv')
train_data_with_dates["ds"] = pd.to_datetime(train_data_with_dates["ds"])

test_data = TabularDataset('X_test_raw.csv')
test_data["ds"] = pd.to_datetime(test_data["ds"])
#test_data
```

Loaded data from: X_train_raw.csv | Columns = 56 / 56 | Rows = 93023 -> 93023

Loaded data from: X_test_raw.csv | Columns = 55 / 55 | Rows = 4608 -> 4608

```
[13]: test_ids = TabularDataset('test.csv')
test_ids["time"] = pd.to_datetime(test_ids["time"])
# merge test_data with test_ids
test_data_merged = pd.merge(test_data, test_ids, how="inner", right_on=["time",
↪ "location"], left_on=["ds", "location"])

#test_data_merged
```

Loaded data from: test.csv | Columns = 4 / 4 | Rows = 2160 -> 2160

```
[14]: # predict, grouped by location
predictions = []
location_map = {
    "A": 0,
    "B": 1,
    "C": 2
}
for loc, group in test_data.groupby('location'):
    i = location_map[loc]
    subset = test_data_merged[test_data_merged["location"] == loc].
↪reset_index(drop=True)
    #print(subset)
    pred = predictors[i].predict(subset)
    subset["prediction"] = pred
    predictions.append(subset)

# get past predictions
```



```

    past_pred = predictors[i].
    ↪predict(train_data_with_dates[train_data_with_dates["location"] == loc])
    train_data_with_dates.loc[train_data_with_dates["location"] == loc,
    ↪"prediction"] = past_pred

```

```

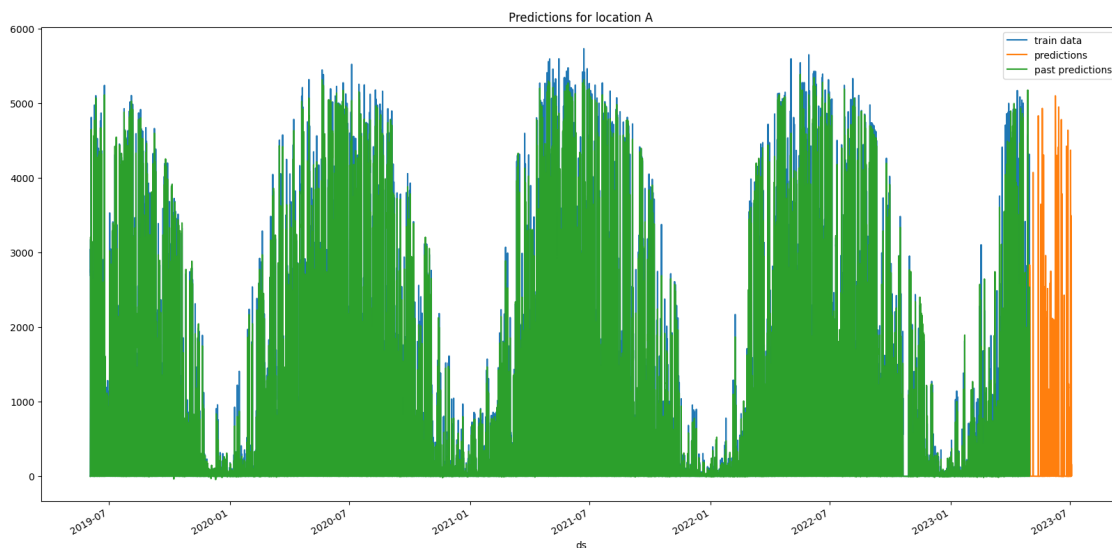
[15]: # plot predictions for location A, in addition to train data for A
for loc, idx in location_map.items():
    fig, ax = plt.subplots(figsize=(20, 10))
    # plot train data
    train_data_with_dates[train_data_with_dates["location"]==loc].plot(x='ds',
    ↪y='y', ax=ax, label="train data")

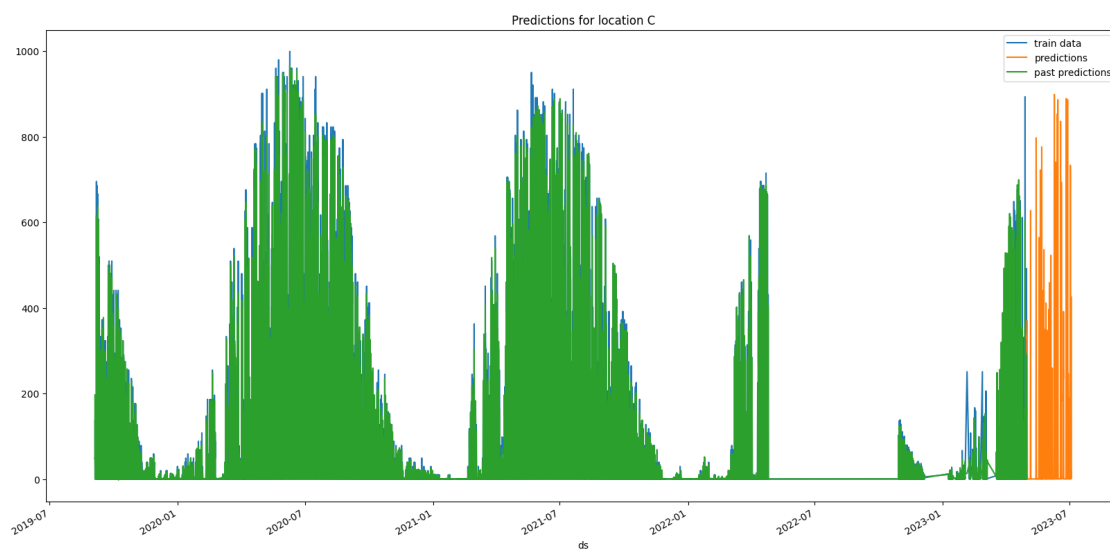
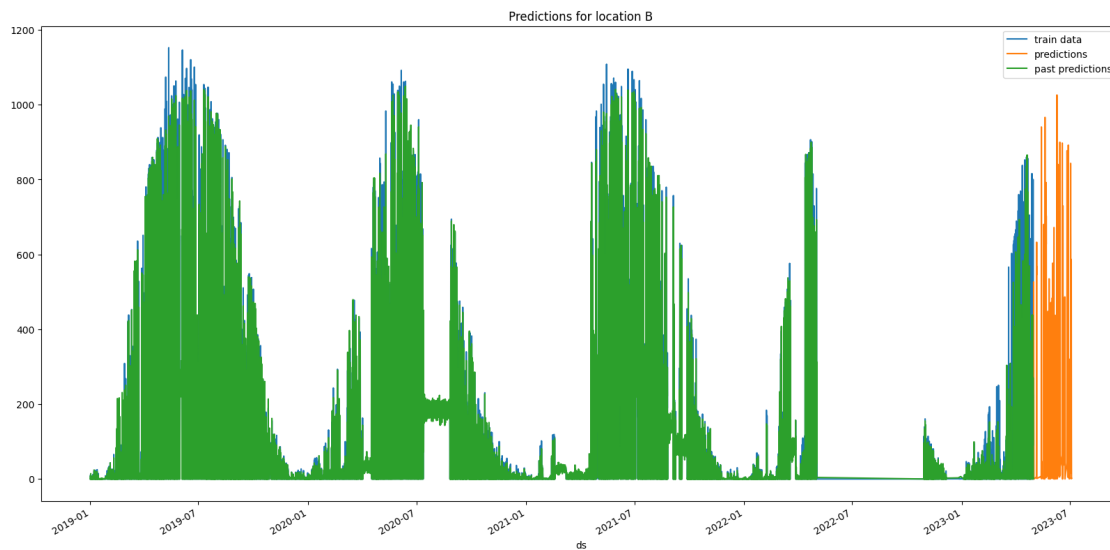
    # plot predictions
    predictions[idx].plot(x='ds', y='prediction', ax=ax, label="predictions")

    # plot past predictions
    train_data_with_dates[train_data_with_dates["location"]==loc].plot(x='ds',
    ↪y='prediction', ax=ax, label="past predictions")

    # title
    ax.set_title(f"Predictions for location {loc}")

```





```
[16]: # concatenate predictions
submissions_df = pd.concat(predictions)
submissions_df = submissions_df[["id", "prediction"]]
submissions_df
```

```
[16]:      id  prediction
0      0    0.119880
1      1    0.116737
2      2    0.181344
3      3   61.310646
```

```

4          4  492.816956
..      ...      ...
715  2155    61.726730
716  2156    34.416271
717  2157     9.828968
718  2158     1.234847
719  2159    18.171741

```

[2160 rows x 2 columns]

```

[17]: # Save the submission DataFrame to submissions folder, create new name based on
      ↪ last submission, format is submission_<last_submission_number + 1>.csv

      # Save the submission
      print(f"Saving submission to submissions/{new_filename}.csv")
      submissions_df.to_csv(os.path.join('submissions', f"{new_filename}.csv"),
      ↪ index=False)
      print("jall1a")

```

Saving submission to submissions/submission_89.csv
jall1a

```

[18]: # save this running notebook
      from IPython.display import display, Javascript
      import time

      # hei123

      display(Javascript("IPython.notebook.save_checkpoint();"))

      time.sleep(3)

```

<IPython.core.display.Javascript object>

```

[19]: # save this notebook to submissions folder
      import subprocess
      import os
      subprocess.run(["jupyter", "nbconvert", "--to", "pdf", "--output", os.path.
      ↪ join('notebook_pdfs', f"{new_filename}.pdf"), "autogluon_each_location.
      ↪ ipynb"])

```

[NbConvertApp] Converting notebook autogluon_each_location.ipynb to pdf
/opt/conda/lib/python3.10/site-packages/nbconvert/utils/pandoc.py:51:
RuntimeWarning: You are using an unsupported version of pandoc (2.9.2.1).
Your version must be at least (2.14.2) but less than (4.0.0).
Refer to <https://pandoc.org/installing.html>.
Continuing with doubts...
check_pandoc_version()
[NbConvertApp] Support files will be in notebook_pdfs/submission_89_files/

```
[NbConvertApp] Making directory
./notebook_pdfs/submission_89_files/notebook_pdfs
[NbConvertApp] Writing 171500 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 1656643 bytes to notebook_pdfs/submission_89.pdf
```

```
[19]: CompletedProcess(args=['jupyter', 'nbconvert', '--to', 'pdf', '--output',
'notebook_pdfs/submission_89.pdf', 'autogluon_each_location.ipynb'],
returncode=0)
```

```
[20]: # feature importance
location="A"
split_time = pd.Timestamp("2022-10-28 22:00:00")
estimated = train_data_with_dates[train_data_with_dates["ds"] >= split_time]
estimated = estimated[estimated["location"] == location]
predictors[0].feature_importance(feature_stage="original", data=estimated,
↪time_limit=60*10)
```

These features in provided data are not utilized by the predictor and will be ignored: ['ds', 'sample_weight', 'location', 'prediction']

Computing feature importance via permutation shuffling for 52 features using 4418 rows with 10 shuffle sets... Time limit: 600s...

5843.05s = Expected runtime (584.31s per shuffle set)

327.5s = Actual runtime (Completed 1 of 10 shuffle sets) (Early stopping due to lack of time...)

```
[20]:
```

	importance	stddev	p_value	n	p99_high	\
direct_rad_1h:J	1.940558e+02	NaN	NaN	1	NaN	
diffuse_rad_1h:J	3.591599e+01	NaN	NaN	1	NaN	
clear_sky_energy_1h:J	2.925134e+01	NaN	NaN	1	NaN	
direct_rad:W	2.535299e+01	NaN	NaN	1	NaN	
sun_azimuth:d	2.089652e+01	NaN	NaN	1	NaN	
clear_sky_rad:W	2.065933e+01	NaN	NaN	1	NaN	
diffuse_rad:W	1.963015e+01	NaN	NaN	1	NaN	
direct_rad_1h:J_not_shifted	1.018332e+01	NaN	NaN	1	NaN	
total_cloud_cover:p	9.883878e+00	NaN	NaN	1	NaN	
wind_speed_u_10m:ms	8.990335e+00	NaN	NaN	1	NaN	
cloud_base_agl:m	7.453726e+00	NaN	NaN	1	NaN	
diffuse_rad_1h:J_not_shifted	5.892141e+00	NaN	NaN	1	NaN	
sun_elevation:d	5.214306e+00	NaN	NaN	1	NaN	
relative_humidity_1000hPa:p	4.554596e+00	NaN	NaN	1	NaN	
visibility:m	4.398705e+00	NaN	NaN	1	NaN	
ceiling_height_agl:m	4.086260e+00	NaN	NaN	1	NaN	

wind_speed_v_10m:ms	3.657893e+00	NaN	NaN	1	NaN
effective_cloud_cover:p	3.359665e+00	NaN	NaN	1	NaN
wind_speed_10m:ms	2.823507e+00	NaN	NaN	1	NaN
precip_type_5min:idx	2.710210e+00	NaN	NaN	1	NaN
is_in_shadow:idx	2.541399e+00	NaN	NaN	1	NaN
is_day:idx	2.178597e+00	NaN	NaN	1	NaN
clear_sky_energy_1h:J_not_shifted	2.109882e+00	NaN	NaN	1	NaN
t_1000hPa:K	1.490996e+00	NaN	NaN	1	NaN
snow_water:kgm2	1.451692e+00	NaN	NaN	1	NaN
msl_pressure:hPa	1.081244e+00	NaN	NaN	1	NaN
pressure_100m:hPa	9.382244e-01	NaN	NaN	1	NaN
sfc_pressure:hPa	9.021291e-01	NaN	NaN	1	NaN
pressure_50m:hPa	8.012686e-01	NaN	NaN	1	NaN
fresh_snow_24h:cm_not_shifted	7.550734e-01	NaN	NaN	1	NaN
snow_melt_10min:mm	6.316097e-01	NaN	NaN	1	NaN
precip_5min:mm	6.019441e-01	NaN	NaN	1	NaN
air_density_2m:kgm3	5.994583e-01	NaN	NaN	1	NaN
fresh_snow_3h:cm_not_shifted	5.557588e-01	NaN	NaN	1	NaN
fresh_snow_6h:cm	5.173812e-01	NaN	NaN	1	NaN
fresh_snow_24h:cm	2.497309e-01	NaN	NaN	1	NaN
elevation:m	2.117588e-01	NaN	NaN	1	NaN
super_cooled_liquid_water:kgm2	1.683345e-01	NaN	NaN	1	NaN
fresh_snow_12h:cm_not_shifted	1.391759e-01	NaN	NaN	1	NaN
snow_depth:cm	1.268459e-01	NaN	NaN	1	NaN
absolute_humidity_2m:gm3	6.457852e-02	NaN	NaN	1	NaN
fresh_snow_1h:cm	5.424152e-02	NaN	NaN	1	NaN
dew_point_2m:K	5.282881e-02	NaN	NaN	1	NaN
rain_water:kgm2	4.742758e-02	NaN	NaN	1	NaN
fresh_snow_3h:cm	3.997476e-02	NaN	NaN	1	NaN
is_estimated	1.726875e-09	NaN	NaN	1	NaN
wind_speed_w_1000hPa:ms	-5.310067e-04	NaN	NaN	1	NaN
fresh_snow_1h:cm_not_shifted	-2.031703e-02	NaN	NaN	1	NaN
dew_or_rime:idx	-2.656923e-02	NaN	NaN	1	NaN
prob_rime:p	-3.239033e-02	NaN	NaN	1	NaN
fresh_snow_6h:cm_not_shifted	-3.706552e-02	NaN	NaN	1	NaN
fresh_snow_12h:cm	-2.250163e-01	NaN	NaN	1	NaN

p99_low

direct_rad_1h:J	NaN
diffuse_rad_1h:J	NaN
clear_sky_energy_1h:J	NaN
direct_rad:W	NaN
sun_azimuth:d	NaN
clear_sky_rad:W	NaN
diffuse_rad:W	NaN
direct_rad_1h:J_not_shifted	NaN
total_cloud_cover:p	NaN

wind_speed_u_10m:ms	NaN
cloud_base_agl:m	NaN
diffuse_rad_1h:J_not_shifted	NaN
sun_elevation:d	NaN
relative_humidity_1000hPa:p	NaN
visibility:m	NaN
ceiling_height_agl:m	NaN
wind_speed_v_10m:ms	NaN
effective_cloud_cover:p	NaN
wind_speed_10m:ms	NaN
precip_type_5min:idx	NaN
is_in_shadow:idx	NaN
is_day:idx	NaN
clear_sky_energy_1h:J_not_shifted	NaN
t_1000hPa:K	NaN
snow_water:kgm2	NaN
msl_pressure:hPa	NaN
pressure_100m:hPa	NaN
sfc_pressure:hPa	NaN
pressure_50m:hPa	NaN
fresh_snow_24h:cm_not_shifted	NaN
snow_melt_10min:mm	NaN
precip_5min:mm	NaN
air_density_2m:kgm3	NaN
fresh_snow_3h:cm_not_shifted	NaN
fresh_snow_6h:cm	NaN
fresh_snow_24h:cm	NaN
elevation:m	NaN
super_cooled_liquid_water:kgm2	NaN
fresh_snow_12h:cm_not_shifted	NaN
snow_depth:cm	NaN
absolute_humidity_2m:gm3	NaN
fresh_snow_1h:cm	NaN
dew_point_2m:K	NaN
rain_water:kgm2	NaN
fresh_snow_3h:cm	NaN
is_estimated	NaN
wind_speed_w_1000hPa:ms	NaN
fresh_snow_1h:cm_not_shifted	NaN
dew_or_rime:idx	NaN
prob_rime:p	NaN
fresh_snow_6h:cm_not_shifted	NaN
fresh_snow_12h:cm	NaN

```
[21]: # feature importance
observed = train_data_with_dates[train_data_with_dates["ds"] < split_time]
observed = observed[observed["location"] == location]
```

```
predictors[0].feature_importance(feature_stage="original", data=observed,
↳time_limit=60*10)
```

These features in provided data are not utilized by the predictor and will be ignored: ['ds', 'sample_weight', 'location', 'prediction']

Computing feature importance via permutation shuffling for 52 features using 5000 rows with 10 shuffle sets... Time limit: 600s...

6627.4s = Expected runtime (662.74s per shuffle set)

371.17s = Actual runtime (Completed 1 of 10 shuffle sets) (Early stopping due to lack of time...)

```
[21]:
```

	importance	stddev	p_value	n	p99_high	\
direct_rad_1h:J	4.109162e+02	NaN	NaN	1	NaN	
clear_sky_energy_1h:J	1.034281e+02	NaN	NaN	1	NaN	
diffuse_rad_1h:J	9.430053e+01	NaN	NaN	1	NaN	
clear_sky_rad:W	9.140538e+01	NaN	NaN	1	NaN	
sun_azimuth:d	8.042408e+01	NaN	NaN	1	NaN	
diffuse_rad:W	6.929680e+01	NaN	NaN	1	NaN	
direct_rad:W	6.092283e+01	NaN	NaN	1	NaN	
ceiling_height_agl:m	3.957188e+01	NaN	NaN	1	NaN	
clear_sky_energy_1h:J_not_shifted	3.513699e+01	NaN	NaN	1	NaN	
cloud_base_agl:m	3.443293e+01	NaN	NaN	1	NaN	
wind_speed_u_10m:ms	3.199988e+01	NaN	NaN	1	NaN	
direct_rad_1h:J_not_shifted	3.061455e+01	NaN	NaN	1	NaN	
total_cloud_cover:p	2.890556e+01	NaN	NaN	1	NaN	
sun_elevation:d	2.816161e+01	NaN	NaN	1	NaN	
wind_speed_v_10m:ms	2.786047e+01	NaN	NaN	1	NaN	
t_1000hPa:K	2.753051e+01	NaN	NaN	1	NaN	
effective_cloud_cover:p	2.734352e+01	NaN	NaN	1	NaN	
relative_humidity_1000hPa:p	2.677388e+01	NaN	NaN	1	NaN	
visibility:m	2.543085e+01	NaN	NaN	1	NaN	
wind_speed_10m:ms	2.478997e+01	NaN	NaN	1	NaN	
diffuse_rad_1h:J_not_shifted	2.145941e+01	NaN	NaN	1	NaN	
dew_point_2m:K	1.549413e+01	NaN	NaN	1	NaN	
absolute_humidity_2m:gm3	1.518191e+01	NaN	NaN	1	NaN	
precip_type_5min:idx	1.483529e+01	NaN	NaN	1	NaN	
air_density_2m:kgm3	1.376474e+01	NaN	NaN	1	NaN	
msl_pressure:hPa	1.138257e+01	NaN	NaN	1	NaN	
pressure_100m:hPa	1.125639e+01	NaN	NaN	1	NaN	
snow_water:kgm2	1.102535e+01	NaN	NaN	1	NaN	
sfc_pressure:hPa	1.097878e+01	NaN	NaN	1	NaN	
pressure_50m:hPa	1.025762e+01	NaN	NaN	1	NaN	
super_cooled_liquid_water:kgm2	7.206754e+00	NaN	NaN	1	NaN	
precip_5min:mm	6.920690e+00	NaN	NaN	1	NaN	
is_day:idx	4.250430e+00	NaN	NaN	1	NaN	
is_in_shadow:idx	3.995840e+00	NaN	NaN	1	NaN	
rain_water:kgm2	3.014453e+00	NaN	NaN	1	NaN	

fresh_snow_24h:cm_not_shifted	1.760273e+00	NaN	NaN	1	NaN
fresh_snow_24h:cm	1.419715e+00	NaN	NaN	1	NaN
snow_depth:cm	6.194939e-01	NaN	NaN	1	NaN
fresh_snow_6h:cm	4.414205e-01	NaN	NaN	1	NaN
dew_or_rime:idx	4.189895e-01	NaN	NaN	1	NaN
fresh_snow_12h:cm_not_shifted	3.796225e-01	NaN	NaN	1	NaN
fresh_snow_12h:cm	3.254430e-01	NaN	NaN	1	NaN
prob_rime:p	2.358120e-01	NaN	NaN	1	NaN
snow_melt_10min:mm	9.727319e-02	NaN	NaN	1	NaN
fresh_snow_3h:cm_not_shifted	9.550108e-02	NaN	NaN	1	NaN
fresh_snow_6h:cm_not_shifted	2.797795e-02	NaN	NaN	1	NaN
is_estimated	1.888275e-08	NaN	NaN	1	NaN
elevation:m	3.814691e-10	NaN	NaN	1	NaN
wind_speed_w_1000hPa:ms	-1.922703e-03	NaN	NaN	1	NaN
fresh_snow_3h:cm	-8.675084e-03	NaN	NaN	1	NaN
fresh_snow_1h:cm	-3.584639e-02	NaN	NaN	1	NaN
fresh_snow_1h:cm_not_shifted	-3.714310e-02	NaN	NaN	1	NaN

p99_low

direct_rad_1h:J	NaN
clear_sky_energy_1h:J	NaN
diffuse_rad_1h:J	NaN
clear_sky_rad:W	NaN
sun_azimuth:d	NaN
diffuse_rad:W	NaN
direct_rad:W	NaN
ceiling_height_agl:m	NaN
clear_sky_energy_1h:J_not_shifted	NaN
cloud_base_agl:m	NaN
wind_speed_u_10m:ms	NaN
direct_rad_1h:J_not_shifted	NaN
total_cloud_cover:p	NaN
sun_elevation:d	NaN
wind_speed_v_10m:ms	NaN
t_1000hPa:K	NaN
effective_cloud_cover:p	NaN
relative_humidity_1000hPa:p	NaN
visibility:m	NaN
wind_speed_10m:ms	NaN
diffuse_rad_1h:J_not_shifted	NaN
dew_point_2m:K	NaN
absolute_humidity_2m:gm3	NaN
precip_type_5min:idx	NaN
air_density_2m:kgm3	NaN
msl_pressure:hPa	NaN
pressure_100m:hPa	NaN
snow_water:kgm2	NaN

sfc_pressure:hPa	NaN
pressure_50m:hPa	NaN
super_cooled_liquid_water:kgm2	NaN
precip_5min:mm	NaN
is_day:idx	NaN
is_in_shadow:idx	NaN
rain_water:kgm2	NaN
fresh_snow_24h:cm_not_shifted	NaN
fresh_snow_24h:cm	NaN
snow_depth:cm	NaN
fresh_snow_6h:cm	NaN
dew_or_rime:idx	NaN
fresh_snow_12h:cm_not_shifted	NaN
fresh_snow_12h:cm	NaN
prob_rime:p	NaN
snow_melt_10min:mm	NaN
fresh_snow_3h:cm_not_shifted	NaN
fresh_snow_6h:cm_not_shifted	NaN
is_estimated	NaN
elevation:m	NaN
wind_speed_w_1000hPa:ms	NaN
fresh_snow_3h:cm	NaN
fresh_snow_1h:cm	NaN
fresh_snow_1h:cm_not_shifted	NaN

```
[22]: display(Javascript("IPython.notebook.save_checkpoint();"))
time.sleep(3)

subprocess.run(["jupyter", "nbconvert", "--to", "pdf", "--output", os.path.
    ↪join('notebook_pdfs', f"{new_filename}_with_feature_importance.pdf"),
    ↪"autogluon_each_location.ipynb"])
```

<IPython.core.display.Javascript object>

```
[NbConvertApp] Converting notebook autogluon_each_location.ipynb to pdf
/opt/conda/lib/python3.10/site-packages/nbconvert/utils/pandoc.py:51:
RuntimeWarning: You are using an unsupported version of pandoc (2.9.2.1).
Your version must be at least (2.14.2) but less than (4.0.0).
Refer to https://pandoc.org/installing.html.
Continuing with doubts...
    check_pandoc_version()
[NbConvertApp] Support files will be in
notebook_pdfs/submission_89_with_feature_importance_files/
[NbConvertApp] Making directory
./notebook_pdfs/submission_89_with_feature_importance_files/notebook_pdfs
[NbConvertApp] Writing 182983 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
```

```
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 1910371 bytes to
notebook_pdfs/submission_89_with_feature_importance.pdf
```

```
[22]: CompletedProcess(args=['jupyter', 'nbconvert', '--to', 'pdf', '--output',
'notebook_pdfs/submission_89_with_feature_importance.pdf',
'autogluon_each_location.ipynb'], returncode=0)
```

```
[23]: # import subprocess

# def execute_git_command(directory, command):
#     """Execute a Git command in the specified directory."""
#     try:
#         result = subprocess.check_output(['git', '-C', directory] + command,
# ↪ stderr=subprocess.STDOUT)
#         return result.decode('utf-8').strip(), True
#     except subprocess.CalledProcessError as e:
#         print(f"Git command failed with message: {e.output.decode('utf-8')}.
# ↪ strip()}"
#         return e.output.decode('utf-8').strip(), False

# git_repo_path = "."

# execute_git_command(git_repo_path, ['config', 'user.email',
# ↪ 'henrikskog01@gmail.com'])
# execute_git_command(git_repo_path, ['config', 'user.name', hello if hello is
# ↪ not None else 'Henrik eller Jørgen'])

# branch_name = new_filename

# # add datetime to branch name
# branch_name += f"_{pd.Timestamp.now().strftime('%Y-%m-%d_%H-%M-%S')}"

# commit_msg = "run result"

# execute_git_command(git_repo_path, ['checkout', '-b', branch_name])

# # Navigate to your repo and commit changes
# execute_git_command(git_repo_path, ['add', '.'])
# execute_git_command(git_repo_path, ['commit', '-m', commit_msg])

# # Push to remote
# output, success = execute_git_command(git_repo_path, ['push',
# ↪ 'origin', branch_name])
```

```
# # If the push fails, try setting an upstream branch and push again
# if not success and 'upstream' in output:
#     print("Attempting to set upstream and push again...")
#     execute_git_command(git_repo_path, ['push', '--set-upstream',  
↪ 'origin',branch_name])
#     execute_git_command(git_repo_path, ['push', 'origin', 'henrik_branch'])
# execute_git_command(git_repo_path, ['checkout', 'main'])
```